

(43) Date of publication of application: 13 . 12 . 94

G06F 9/46  
G06F 9/44

(71) Applicant: **PERSONAL JOHO KANKYO  
KYOKAI**

(72) Inventor: UYAMA MASASHI

(57) Abstract:

Figure 1 is a block diagram illustrating a multi-layered architecture for a multi-agent system. The architecture consists of several layers and components:

- User Layer (101):** Contains the text "In the News Paper, News Article ID3238 was Recalled by...".
- Function Level (201):** A layer above the User Layer.
- User's Task Level (202):** A layer above the Function Level.
- Semantic Level (203):** A layer above the User's Task Level.
- Interaction Level (204):** A layer above the Semantic Level. It contains a "Metaphor" box with the text "Place News Article ID3238", "Completed", and "News Paper".
- Task-Phase Description Layer (TPD Layer) (205):** A layer above the Interaction Level. It contains a "Server Task" box with the text "Place News Article ID3238", "Completed", and "News Paper".
- Single Cast Layer (103):** A layer above the TPD Layer. It contains the text "(v.1) (0, "Context", resource-id, sender-id, message-on, TPD)".
- Broadcast Layer (104):** A layer above the Single Cast Layer. It contains the text "(v.1.2) ("Context", "Resource", can reach-id-3, "Completed", "News Paper", "Place News Article ID3238")".

The architecture is connected to three external components:

- AGENTS (102):** Connected to the Interaction Level via "Stored Percepts" and "Metaphor".
- Studio Manager (106):** Connected to the Interaction Level via "Server Task" and "Completed".
- Bid Arbitrator (107):** Connected to the Broadcast Layer via "Completed".

The entire system is labeled 100.

**CONSTITUTION:** The studio management agent 106 manages a history of task state descriptions sent out to a studio 100 as context information. The bid arbiter agent 107 performs bid evaluation, by contract net protocol by using the stored context information. An interactive manager agent 105 explains the process of a context dependent process to a user and assists the customization of this process by the user. Agent groups 105-107 send and receive task states through the studio 100 by using a studio function group and the bid arbiter agent 107 evaluates the bid according to the context that the studio management agent 106 manages.

COPYRIGHT: (C)1994,JPO

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平6-342375

(43) 公開日 平成6年(1994)12月13日

(51) Int.Cl.<sup>5</sup>

G 0 6 F 9/46  
9/44

識別記号

3 4 0 A  
5 5 2

庁内整理番号

8120-5B  
9193-5B

F I

技術表示箇所

審査請求 未請求 請求項の数 8 O L (全 26 頁)

(21) 出願番号 特願平5-5602

(22) 出願日 平成5年(1993)1月18日

特許法第30条第1項適用申請有り 平成4年7月21日、  
財団法人パーソナル情報環境協会発行の「第4回 F R I  
E N D 21 成果発表会論文集」に発表

(71) 出願人 591067510

財団法人パーソナル情報環境協会  
東京都港区虎ノ門1丁目17番1号

(72) 発明者 宇山 政志

東京都港区虎ノ門1-17-1 財団法人パー  
ソナル情報環境協会内

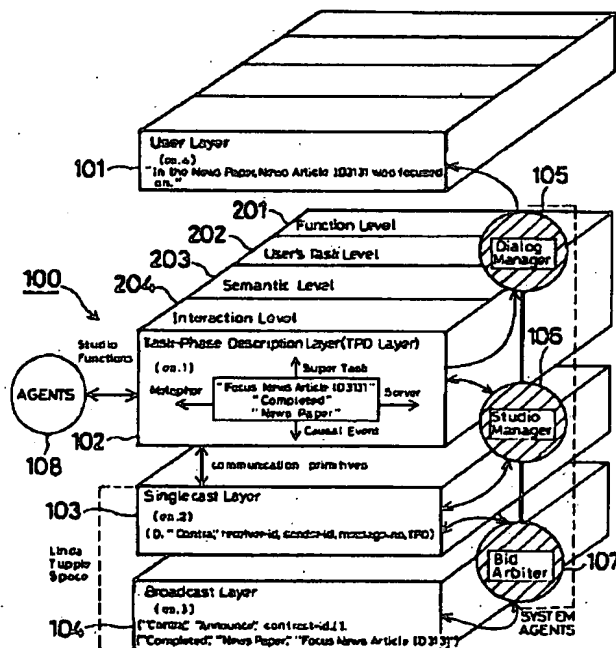
(74) 代理人 弁理士 阿部 龍吉 (外7名)

(54) 【発明の名称】 ヒューマンインタフェースのアーキテクチャモデル

(57) 【要約】 (修正有)

【目的】 カスタマイズされたタスクモデルに新たに追加された部分タスクモデルを統合することを可能にする。

【構成】 スタジオに送出されたタスク状態記述の履歴をコンテキスト情報として管理するスタジオ管理エージェント106と、蓄積したコンテキスト情報を用いて契約ネットプロトコルにおける入札評価する入札アービタエージェント107と、コンテキスト依存処理の経過をユーザに説明し、ユーザにコンテキスト依存処理をカスタマイズする手段を提供する対話マネージャエージェント105とを有し、エージェント群がスタジオを介してスタジオ関数群を用いてタスク状態記述を授受し、スタジオ管理エージェント106の管理するコンテキストに基づき入札アービタエージェント107が入札を評価することにより、モジュール群が相互のコンテキストに応じて動作するように構成した。



BEST AVAILABLE COPY

## 【特許請求の範囲】

【請求項1】 独立性を持った要素機能に分割された複数のエージェントと共有媒体として種々の情報を読み書きするスタジオとを備えたアーキテクチャモデルにおいて、スタジオに送出されたタスク状態記述の履歴をコンテキスト情報として管理するスタジオ管理エージェントと、蓄積したコンテキスト情報を用いて契約ネットプロトコルにおける入札評価をする入札アービタエージェントと、コンテキスト依存処理の経過をユーザに説明し、ユーザにコンテキスト依存処理をカスタマイズする手段を提供する対話マネージャエージェントとを有し、エージェント群がスタジオを介しスタジオ関数群を用いてタスク状態記述を授受し、スタジオ管理エージェントの管理するコンテキストに基づき入札アービタエージェントが入札を評価することにより、モジュール群が相互のコンテキストに応じて動作するように構成したことを特徴とするヒューマンインタフェースのアーキテクチャモデル。

【請求項2】 請求項1記載のヒューマンインタフェースのアーキテクチャモデルにおいて、コンテキストに依存した条件分岐を契約ネットプロトコルで行い、契約アナウンス及び入札の形式を公開することで、新たな機能をユーザに提示するエージェントを設計、追加できるように構成したことを特徴とするヒューマンインタフェースのアーキテクチャモデル。

【請求項3】 請求項1記載のヒューマンインタフェースのアーキテクチャモデルにおいて、メタファ環境エージェントとファンクションエージェントと部分タスクエージェントを備えたことを特徴とするヒューマンインタフェースのアーキテクチャモデル。

【請求項4】 請求項1記載のヒューマンインタフェースのアーキテクチャモデルにおいて、タスク状態記述は、タスク、タスクの状態、及びメタファの種類／機能の種類を示す3つの文字列並びに2つのエージェントへのポインタ、2つのタスク状態記述へのポインタからなることを特徴とするヒューマンインタフェースのアーキテクチャモデル。

【請求項5】 請求項1記載のヒューマンインタフェースのアーキテクチャモデルにおいて、スタジオ関数群として、ウインドウレベルの操作を表すタスク状態記述、メタファ環境エージェントと部分タスクエージェントとの間の契約アナウンス及び部分タスクエージェントの実行状態を表すタスク状態記述、メタファ環境エージェントと部分タスクエージェントとの間で対話を行うタスク状態記述、ファンクションエージェントと部分タスクエージェントとの間で対話を行うタスク状態記述に応じた関数が用意されていることを特徴とするヒューマンインタフェースのアーキテクチャモデル。

【請求項6】 請求項1記載のヒューマンインタフェースのアーキテクチャモデルにおいて、契約の入札は、契

約に成功したとき実行するタスクを示すプランとコンテキスト条件の列の形式で行うことを特徴とするヒューマンインタフェースのアーキテクチャモデル。

【請求項7】 請求項1記載のヒューマンインタフェースのアーキテクチャモデルにおいて、入札の評価は、各入札のコンテキスト条件列とタスク状態記述履歴とのパターンマッチングにより行われることを特徴とするヒューマンインタフェースのアーキテクチャモデル。

【請求項8】 請求項1記載のヒューマンインタフェースのアーキテクチャモデルにおいて、複数のエージェント間にわたるコンテキスト依存処理をユーザの可読な形で提示し、ユーザの想定したコンテキストとベンダの想定したコンテキストで不一致がおこったとき、ユーザがシステムに対し割り込みにより意思表示を行うと共に、システムに正しいコンテキストを教示できるようにしたことを特徴とするヒューマンインタフェースのアーキテクチャモデル。

## 【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、独立性を持った要素機能に分割された複数のエージェントと共有媒体として種々の情報を読み書きするスタジオとを備えたヒューマンインタフェースのアーキテクチャモデルに関する。

【0002】

【従来の技術】 一般に、コンピュータシステムは、異なる設計者が作成した複数のソフトウェアモジュールから構成され、新たに設計されたソフトウェアモジュールをシステムに追加することは、システムの能力を改善し、ユーザ自身のタスク遂行能力を向上させる可能性を持つ。しかし、新たなモジュールの追加は、それ以前にユーザとシステムの間に形成されていた合意を破壊するという問題を引き起こす。

【0003】 ユーザがコンピュータを滞りなく利用している時には、ユーザとシステムの間に何らかの合意が成立していると考えられる。ユーザとシステムの間の合意とは、①ユーザのタスクとシステムの機能とのマッピング、つまり、ユーザがタスク遂行上解決すべき多くのゴールのうちの、どのゴールを解決するために、システムの提供する多くの機能のうちのどの機能を用いるか、②ゴールを解決するためにユーザがシステムに対して行うべき操作、③ユーザの操作に応じてシステムが提示すべきフィードバック、に関してユーザが保有する知識、及びシステムの保有する知識の双方を指す。

【0004】 合意の一部としてシステムが保有する知識をタスクモデルと呼ぶ。タスクモデルを利用することで、システムは、ユーザのタスク遂行のコンテキストを把握し、ユーザに対する知的な助力を行うことができる。例えば、ルーチンタスクの自動化、ユーザの意思決定が必要な場面での選択肢の提示、意味的に不適切な行為の認識・教示、などを可能にしている。

【0005】タスクモデルの一部は、システム設計者が、ユーザのタスクを分析、作成することにより提供される。また、機能モジュール群のベンダは、そのモジュール群を用いて、ユーザがゴールを解決する手順を想定し、記述することができ、これらの記述をタスクモデルの一部として、システムに提供することができる。

【0006】しかし、一般に、ユーザは異なるベンダの作成した複数のツールを用いてタスクを遂行するため、個々のベンダがユーザのタスク全体を予期することはできず、各ベンダの提供するタスク遂行に関する知識は、ユーザのタスクの一部をモデル化しているにすぎない。また、ある時点のタスク分析から得られた知識だけでは、新たなモジュールの追加や、それに伴うユーザのタスク自体の変化に対応していくことができない。

【0007】このため、ベンダの設計した知識や、タスク分析によって得られる知識は、異なるベンダの作成した複数のツールを用いてタスクを遂行する個々のユーザにとって、自分のタスクを適切に反映したタスクモデルではなく、ユーザのタスクの一部をモデル化した部分タスクモデルである。部分タスクモデルを利用したシステムの例としては、POISEシステムがある。

【0008】システム設計者や、個々のベンダにより与えられた部分タスクモデル群を、実際のユーザのタスク遂行に合わせて適切に組み合わせ、修正することで初めてタスクモデルが形成される。タスクモデルの生成のためには、ユーザが部分タスクモデル群をカスタマイズするための機能、または、システムが自動的に部分タスクモデルを修正する学習機能が必要である。部分タスクモデルからタスクモデルを生成するためのカスタマイズ機能を提供するシステムの例としては、HP NewWave環境、HP SoftBench環境がある。

【0009】コンピュータシステムを機能モジュール群とインタフェースアーキテクチャの2つの部分に分割して考えると、機能モジュール群は、ベンダが想定したタスクを解決するために、ベンダにより設計提供されたモジュール群であり、インタフェースアーキテクチャは、タスクモデル、部分タスクモデルなどの知識を管理し、ユーザとシステムの間で合意を形成する役割を果たすソフトウェアである。

【0010】一般に、コンピュータシステムは、異なる設計者が作成した複数のソフトウェアモジュールから構成され、新たに設計されたソフトウェアモジュールをシステムに追加することは、システムの能力を改善し、ユーザ自身のタスク遂行能力を向上させる可能性を持つ。しかし、新たなモジュールの追加は、それ以前にユーザとシステムの間で形成されていた合意を破壊するという問題を引き起こす。

【0011】この合意は、ユーザ自身の学習、システムによるユーザへの適応、ユーザによるシステムのカスタマイズ等により形成される。新しいモジュールを利用す

るために、ユーザがこれらの知識を変更し構成し直す必要が生ずる。このようにユーザが知識を変更する必要性が生じることを合意の破壊という。

【0012】図17は部分タスクモデルのカスタマイズによる合意の形成を説明するための図、図18は新しいモジュールの追加に伴う合意の破壊を説明するための図である。ベンダA、ベンダBにより機能モジュール群が実装されたシステムを示したのが図17であり、システムには、同時に機能モジュール群を利用したタスク遂行をモデル化した部分タスクモデルPTMA、PTMBがそれぞれのベンダにより供給されている。ユーザのタスク遂行を的確に支援するためには、ユーザのタスク要求を解決するためにモジュールがいかなる組み合わせで用いられるかの知識が必要である。これらの知識は、ユーザ自身がカスタマイズにより定義するか、インタフェースアーキテクチャが発見する必要がある。

【0013】図17に示すユーザとシステムの間で合意で成立している状況に、ベンダCにより新しい機能モジュール群及び部分タスクモデルが提供されたケースを示したのが図18である。この場合、既存の機能モジュールの利用法が公開されていればベンダCは新たに追加するモジュール群と既存のモジュール群の両方を用いたタスク遂行をモデル化できる。

【0014】ユーザのタスク要求のうち、ベンダCの提供する部分タスクモデルで解決可能な部分が、今図18に示すようにカスタマイズしたタスクモデルで解決可能な部分を包含していたとする。このような場合、従来のシステムでは、ユーザは、

①ベンダCの提供した機能、部分タスクモデルを無視してシステムとユーザの間で合意に基づくタスク遂行を維持する。

【0015】②システムとユーザの間に形成された合意を廃棄し、ベンダCの提供する部分タスクモデルに従ったタスク遂行を受け入れる。この場合、ユーザは、システムの操作法やフィードバックに関する知識を変更する必要がある。

【0016】③カスタマイズしたタスクモデルを新たに追加されたモジュールに適用可能なようにユーザ自身が変更する。

【0017】のいずれかの手法を選択するしかなかった。これらの要求に照らしたとき、上記のように複数のモジュールにわたるユーザのタスク遂行をモデル化した知識を持つことができる従来のシステムはいずれも問題がある。

【0018】〔従来例：POISE〕部分タスクモデルを利用したシステムには、例えばPOISEシステムがある。POISEでは、特定のツールの集合を用いたオフィスワークに対して分析、作成された知識が、システム設計者によってシステムのデータベースとして与えられている。POISEは、このデータベースを用いてオフィスワークを支

援することができる。

【0019】〔POISEのタスク記述〕POISEの知識は、オフィスワークの分析によって獲得され、システム設計者によって、システムのデータベース中の「タスク記述」の集合として与えられている。タスク記述は、正規表現を用いて、高次のタスクを「より低次のタスク」または「ツールの起動」によって定義する形式で記述される。このような情報をデータベース中に持つことで、POISEは、ユーザの操作のコンテキストを認識し、そのコンテキストに応じた動作を計画することができる。例えばユーザが特定のタスクを指定すれば、システムは、実行ステップを計画し、ツールを自動的に起動したり、必要時にユーザに問い合わせたりできる。

【0020】〔POISEシステムの問題〕しかし、POISEでは、知識が特定のツール群を用いたオフィスワークを対象に作成されているため、この環境に新たなツールを加えることや、新たなタスク記述を追加することが考慮されていない。新たなツールや新たなタスク記述が、既存の環境に適切に統合されるためには、その設計者が、既存ツールの起動法や、タスク名、オブジェクト名などについて知っている必要がある。POISEシステムでは、設計者にこれらの情報を公開する単位としてのプロトコルが用意されておらず、また、設計者の追加設計を支援するためのツールも供給されていない。このため、新しいツールやタスク記述の設計者は、現在のデータベース中にある全ての記述を理解しなければならないという問題があった。

【0021】〔従来例：HP NewWave環境とHP SoftBench環境〕部分タスクモデルからタスクモデルを生成するためのカスタマイズ機能を持つシステムの例として、HP NewWave環境、HP SoftBench環境がある。

【0022】〔HP NewWave環境のAgent Task Language〕HP NewWave環境では、Agent Task Language というスクリプト言語をユーザが記述することで、異なるベンダの作成した複数のツールに渡る一連の操作をシステムに登録することができる。これにより、この一連の操作を、アイコンのドラッグという単一の操作で置き換えるという合意がユーザとシステムの間で形成できる。

【0023】〔HP NewWave環境の問題点〕しかし、Agent Task Language でユーザが作成したスクリプトは、一般にベンダが知ることはできない。このため、新しいツールの追加に際し、新規追加モジュールの機能を利用するようにスクリプトを変更することは、ユーザ自身の手作業で行うしかなく、ユーザの負荷が高くなりユーザが新しいツールの機能を利用する障害となるという問題がある。

【0024】〔HP SoftBench環境のツールプロトコルとEncapsulator記述言語〕HP SoftBench環境は、POISEよりもさらに統合された環境であり、より深いレベルのツール間の協調機構を提供している。HP SoftBench環境で

は、機能モジュールの種類毎に公開されるツールプロトコルが公開されており、これが部分タスクモデルと見做せる。ユーザは、Encapsulator記述言語と呼ばれる記述言語により、複数のツールの利用方法をカスタマイズし、タスクモデルを作ることができる。例えばコンパイラモジュールがコンパイルを終了したら、メインモジュールを起動し、コンパイル終了をプログラマチームのメンバーに知らせるようなタスクを記述することが可能である。

10 【0025】〔HP SoftBench環境の問題〕図19はHP SoftBench環境の問題点を説明するための図である。HP SoftBench環境では、ツールプロトコルとして、機能モジュールの利用法が公開されている。これにより図19

(1)のように、新しく機能モジュール群Bを設計したベンダBは、既存の機能モジュール群Aの機能を利用する部分タスクモデルBを設計することができる。しかし、ツールプロトコルを公開するだけでは、図19

20 (2)のように、個々のユーザのタスクモデルを、新規追加の機能モジュール群Bの機能を利用するように修正することはできない。このため、ユーザは、それまでのタスクモデルに固執するか、ベンダBの提供する部分タスクモデルを採用するか、あるいは、ユーザ自身が、Encapsulator記述言語のプログラミングによりタスクモデルを再設計するしかなかった。

【0026】〔従来例：先行特許〕本出願人が既に提案したヒューマンインタフェースのアーキテクチャモデル（特願平3-172823号）では、上記HP SoftBench環境の持つ問題は部分的に解決される。つまり、新しい機能モジュールの追加に際して、それまでのタスクモデルを生かし、新しい機能も利用できる新たなタスクモデルを生成することができる。

【0027】例えば「前回新聞を見てから日付が更新していれば、新しい情報を提示し、そうでなければ以前に見ていた新聞で見ていた情報を提示する」というタスクモデルが存在する環境に対して、ベンダが新たなモジュール群を追加することにより、「前回新聞あるいは番組を見てから日付が更新されていれば、新しい情報が新聞で提示され、そうでない場合には、直前に番組で情報を見ていれば、番組で見ていた情報を含む新聞が提示され、新聞で情報を見ていれば、以前に新聞で見ていた情報を提示する」ように、それまでのタスクモデルを生かして「番組で見ていた情報を新聞で見る」という新しい機能も利用できる新たなタスクモデルを生成することができる。

【0028】ヒューマンインタフェースのアーキテクチャモデルに関して、既に特願平1-222498号（特開平3-84652号公報参照）により本出願人が提案している。このヒューマンインタフェースのアーキテクチャモデルによれば、黒板モデルにおける黒板に相当する共有場をスタジオ、スタジオを介して通信を行うモジ

ユーザ群をエージェントと呼び、複数のエージェントにより要素機能に分割されヒューマンインタフェースに係わる処理や情報を担い、1つのスタジオにより協調動作する複数のエージェントの間のコミュニケーション、同期、内部データの共有を行う。

【0029】また、各エージェントは、独立性をもった機能モジュールであり、スタジオの状態により起動することが可能であり、一つの目的のために一つ以上のエージェントが起動される。このモデルでは、1つのスタジオが全てのエージェントの共有媒体となっていた。そのため、新たなエージェントを追加しようとすると、追加されるエージェントが他の全てのエージェントの動作に副作用を与える可能性が生じるという問題があった。しかも、既存のエージェント群の動作の特定部分にのみ影響を与えることを目的としたエージェントを新たに設計する際にも、他の全てのエージェントの仕様を考慮する必要があり、設計に要するコストが高くなるという問題があった。

【0030】そこで本出願人は、さらに既存のエージェント群の全ての詳細な仕様、動作を意識することなく新たなエージェントを設計できるようにするヒューマンインタフェースのアーキテクチャモデルを提案（特願平3-172823号）した。このモデルは、以下のようなものである。

【0031】図20は先行発明のヒューマンインタフェースのアーキテクチャモデルの概念図である。ファンクションエージェント群3は、ユーザのタスクを実行するための機能と該機能を他のエージェントから利用するための手続きを規定したファンクションプロトコルとを備えたものである。メタファ操作モデルエージェント群1は、ユーザのタスクを実行する際に必要なユーザとのインタラクションを行うための外見と該外見を他のエージェントから利用するための手続きを規定したメタファプロトコルとを備えたものである。メタファ世界モデルエージェント群2は、メタファプロトコルとファンクションプロトコルとの相互変換を行いファンクションエージェント群3の提供する機能の全部あるいは一部をメタファ操作モデルエージェント群1から利用できるようにするものである。各プロトコル毎の部分スタジオ5、6は、共有媒体としてのスタジオを分割したものである。必須エージェント群4は、全てのエージェント群1～3に共通の機能を備え全ての部分スタジオ5、6を参照可能にしたものである。

【0032】先行発明のヒューマンインタフェースのアーキテクチャモデルでは、ファンクションエージェント群3、メタファ操作モデルエージェント群1、メタファ世界モデルエージェント群2、必須エージェント群4からなる全てのエージェント群がスタジオにつながり、各エージェント群1～4がプロトコル毎に設定された部分スタジオ5、6を介して通信を行ってユーザのタスクを

分散、協調して遂行する。

【0033】各エージェント群1～3において、エージェントの送出するメッセージは、部分スタジオ5、6のうちの、該メッセージが属するプロトコルに対応した一つにのみ送出され、エージェント群1～3のうちの該部分スタジオに接続されているエージェント及び必須エージェント群4以外からの参照アクセスはできない。

【0034】部分スタジオ5、6に送出された或るメッセージを部分スタジオ5、6に接続された複数の同一種類のエージェントが受け取り可能な場合、このメッセージを契約のアナウンスメッセージといい、アナウンスメッセージを送信したエージェントを契約のマネージャと呼ぶ。

【0035】アナウンスメッセージを受け取り可能な複数のエージェントは、契約の入札メッセージを契約のマネージャに送信し、マネージャは、これらの入札メッセージをスタジオ履歴エージェントに問い合わせをすることで評価し、最も適当なエージェントに対して落札の通知メッセージを出すことでメッセージの最終的な受け取り手を一意に決定する。

【0036】この契約のアナウンスメッセージ、入札メッセージ、入札の評価方法、落札の通知メッセージは、メタファプロトコルやファンクションプロトコルの一部として公開されている。

【0037】図21は先行発明に係るヒューマンインタフェースのアーキテクチャモデルの1実施例を示す図、図22～図24は先行発明に係るヒューマンインタフェースのアーキテクチャモデルの動作を説明するための図である。

【0038】図21において、新聞メタファ操作モデルエージェント1-1は、新聞メタファを他のエージェントから利用するための手続きを規定した新聞メタファプロトコルを持つメタファ操作モデルエージェントであり、ユーザとのインタラクションを新聞のような外見で行うものである。

【0039】DBエージェント12は、オンラインデータベース機能を他のエージェントから利用するための手続きを規定したDBファンクションプロトコルを持つファンクションエージェントであり、他のエージェントにオンラインデータベースの機能を提供するものである。

【0040】新聞メタファ世界Aエージェント13及び新聞メタファ世界Bエージェント14は、新聞メタファプロトコルに基づき新聞メタファ操作モデルエージェント11と通信を行い、DBファンクションプロトコルに基づきDBエージェント12と通信を行うことで、DBエージェント12の提供する機能を新聞のような外見で利用できるように変換するメタファ世界モデルエージェントである。

【0041】これら2つの新聞メタファ世界モデルエージェントは、DBエージェント12の提供する機能の異

なる部分を利用するものであり、例えば新聞メタファ世界Aエージェント13は、ユーザに対して最新の情報を提示し、新聞メタファ世界Bエージェント14は、ユーザが以前に新聞の外見で見ていた情報を提示する。

【0042】スタジオ履歴エージェント15は、スタジオに送出されるあらゆるメッセージを蓄積し、他のエージェントからの問い合わせに対して、蓄積したメッセージ履歴を走査することで応答するものである。

【0043】上記実施例における新聞メタファプロトコルでは、以下のような契約のためのメッセージ群が定義されている。例えばメッセージ

〔新聞を見る〕

は、メタファ操作モデルエージェント群の新聞メタファ操作モデルエージェント11からメタファ世界モデルエージェント群の各新聞メタファ世界エージェント13、14へ送出される契約のアナウンスメッセージである。この契約のアナウンスメッセージに対しメッセージ形式

〔入札α〕

は、メタファ世界モデルエージェント群の各新聞メタファ世界エージェント13、14から送出される入札メッセージの形式を示す。ここで、αは任意のメッセージボタンである。

【0044】入札評価手続きでは、アナウンスメッセージを送信した新聞メタファ操作モデルエージェント11（契約のマネージャ）が各入札メッセージ中のボタンαのうち、最も最近スタジオ内に送出されたボタンをスタジオ履歴エージェント15に問い合わせる。

【0045】スタジオ履歴エージェント15は、問い合わせに対して時系列に蓄積したメッセージの履歴を遡り、これらのボタンとのマッチングを行い、最初に一致したボタンを契約のマネージャである例えば新聞メタファ操作モデルエージェント11に返答する。

【0046】新聞メタファ操作モデルエージェント11は、該ボタンを入札したメタファ世界モデルエージェント13又は14に落札通知のためのメッセージ

〔新聞を開く〕

を送信する。

【0047】一定時間内にスタジオ履歴エージェント15からの返答がない場合には、ユーザにこれらの入札を提示し選択させる。

【0048】上記の動作を図22～図24によりさらに詳述する。ユーザが新聞のアイコンをクリックすると、図22（イ）に示すように新聞メタファ操作モデルエージェント11が契約のアナウンスメッセージ

〔新聞を見る〕

を部分スタジオに送出し、2つのエージェント、新聞メタファ世界Aエージェント13及び新聞メタファ世界Bエージェント14がこれを受け取る。

【0049】これに対して図22（ロ）に示すように、新聞メタファ世界Aエージェント13は、ユーザに対し

て最新の情報を提示するためのエージェントであるので、必須エージェントである時計エージェント16が午前0時に送出するメッセージにマッチするボタンを含む〔入札〔今日は \*年\*月\*日です〕〕

のような入札メッセージを送出する。同様に、新聞メタファ世界Bエージェント14は、ユーザが以前に新聞の外見で見ていた情報を提示するためのエージェントであるので、新聞メタファ操作モデルエージェント11へのメッセージボタンを含む

〔入札〔画面に 新聞を 表示する〕〕

のような入札メッセージを送出する。

【0050】そこで、新聞メタファ操作モデルエージェント11は、図23（イ）に示すように契約のマネージャとしてスタジオ履歴エージェント15に対しこれらの入札に含まれるボタンのうち、どれが最も最近送出されているかを問い合わせる。

【0051】スタジオ履歴エージェント15は、遡って時系列に蓄積されたメッセージとのボタンマッチングを行い、図23（ロ）に示すように最初にマッチしたボタン、例えば〔今日は \*年\*月\*日です〕を新聞メタファ操作モデルエージェント11に返す。

【0052】これを受けた新聞メタファ操作モデルエージェント11は、図24に示すようにそのボタンを入札した、例えば新聞メタファ世界Aエージェント13に落札通知のためのメッセージ

〔新聞を開く〕

を送信する。

【0053】この入札により、前回新聞を観てから日付が更新していれば、新しい情報を提示し、そうでなければ以前に新聞を観ていた情報を提示する。

【0054】上記の実施例にDBエージェント12の提供する機能をテレビ番組の外見で利用するためのエージェント群を追加した例を示したのが図25である。この場合、先行発明によれば、設計者は、メタファ操作モデルエージェントとして、テレビ番組の外見を他のエージェントから利用するための手続きを規定した番組メタファプロトコルを持つ番組メタファ操作モデルエージェント19を設計し、番組メタファプロトコルとDBファンクションプロトコルのみを参照することで、メタファ世界モデルエージェントとして、番組メタファ世界エージェント21を設計すればよい。このとき、新聞メタファプロトコルを介して通信を行っている新聞メタファ世界Aエージェント13、新聞メタファ世界Bエージェント14、新聞メタファ操作モデルエージェント11について考える必要はない。

【0055】さらに、設計者は、新聞メタファプロトコルとDBファンクションプロトコルのみを参照することで、既存のエージェントである新聞メタファ世界Aエージェント13、新聞メタファ世界Bエージェント14、新聞メタファ操作モデルエージェント11の詳細につい

て考えることなしに、既存の新聞の動作を変更するようなエージェント、新聞メタファ世界Cエージェント20を設計することもできる。

【0056】これにより新聞メタファ世界Cエージェント20は、入札に参加し、ユーザが直前に番組で情報を見ていた場合に、番組で提示されていた情報を含む新聞を提示することができる。これらの動作例を示したのが図26～図28であり、以下にその動作例を説明する。

【0057】まず、ユーザにより新聞のアイコンをクリックされると、図26(イ)に示すように新聞メタファ操作モデルエージェント11は、契約アナウンスメッセージ

【新聞を見る】

を部分スタジオに送出し、3つのエージェント、新聞メタファ世界Aエージェント13、新聞メタファ世界Bエージェント14、新聞メタファ世界Cエージェント20がこれを受け取る。

【0058】これに対して、図26(ロ)に示すように新聞メタファ世界Aエージェント13は、ユーザに対して最新の情報を提示するためのエージェントであるので、必須エージェントである時計エージェント16が午前0時に送出するメッセージにマッチするボタンを含む

【入札【今日は \*年\*月\*日です】】

のような入札メッセージを送出する。同様に、新聞メタファ世界Bエージェント14は、ユーザが以前に新聞の外見で見ていた情報を提示するためのエージェントであるので、新聞メタファ操作モデルエージェント11へのメッセージボタンを含む

【入札【画面に 新聞を 表示する】】

のような入札メッセージを送出する。また、新聞メタファ世界Cエージェント20は、ユーザが直前に番組で見ていた情報を提示するためのエージェントであるので、番組メタファ操作モデルエージェント19へのメッセージボタンを含む

【入札【画面に 番組を 表示する】】

のような入札メッセージを送出する。

【0059】これらの入札メッセージより新聞メタファ操作モデルエージェント11は、図27(イ)に示すようにスタジオ履歴エージェント15に対してこれらの入札に含まれるボタンのうち、どれが最も最近送出されているかを問い合わせる。

【0060】スタジオ履歴エージェント15は、この問い合わせに対し図27(ロ)に示すように遡って時系列に蓄積されたメッセージとのパターンマッチングを行い、最初にマッチしたボタン、例えば【画面に 番組を 表示する】を新聞メタファ操作モデルエージェント11に返す。

【0061】新聞メタファ操作モデルエージェント11は、図28に示すようにそのボタンを入札した、新聞メタファ世界Cエージェント20に落札通知のためのメッ

セージ

【新聞を開く】

を送信する。

【0062】上記のように新聞メタファ世界Cエージェント20を追加することにより、前回新聞、あるいは番組を見てから日付が更新されていれば、新しい情報が新聞で提示され、そうでない場合には、直前に番組で情報を見ていれば、番組で見ていた情報を含む新聞が提示され、新聞で情報を見ていれば、以前新聞で見ていた情報が提示されるように動作を変更することができる。

【0063】【先行特許の問題点】このようにヒューマンインタフェースのアーキテクチャモデル(特願平3-172823号)では、契約ネットのメカニズムによりユーザ自身がタスクモデルを新しいモジュール用に変更する負荷を、ベンダとアーキテクチャによりある程度代行することができる。ベンダが設計した入札メッセージを、部分タスクモデルとして考えることができる。しかし、ベンダの記述とアーキテクチャによる入札評価だけでは、新しい機能でも、ユーザにとって望ましくない機能が実行されてしまう可能性がどうしても残る。このため、ユーザが必要最小限の負荷でカスタマイズできる必要がある。しかし、この先行特許では、ユーザの契約の入札をカスタマイズする機構が用意されていないという問題がある。また、入札の評価手続きを各エージェント毎に設計できるため、システムの動作がユーザにとって一貫しないという問題がある。また、エージェントがメッセージの送信をするための方法、メッセージの共通形式が定義されていないという問題があった。

【0064】

【発明が解決しようとする課題】従来のシステムでは、ユーザは、

①ベンダCの提供した機能、部分タスクモデルを無視してシステムとユーザの間の合意に基づくタスク遂行を維持する。

【0065】②システムとユーザの間に形成された合意を廃棄し、ベンダCの提供する部分タスクモデルに従ったタスク遂行を受け入れる。この場合、ユーザは、システムの操作法やフィードバックに関する知識を変更する必要がある。

【0066】③カスタマイズしたタスクモデルを新たに追加されたモジュールに適用可能なようにユーザ自身が変更する。

【0067】のいずれかの手法を選択するしかなかった。

【0068】先行特許(特願平3-172823号)は、この問題を部分的に解決するが、ユーザが契約の入札をカスタマイズする機構が用意されていない、システムの動作がユーザにとって一貫しない、エージェントがメッセージの送信をするための方法、メッセージの共通形式が定義されていないといった問題があった。

【0069】本発明は、上記の課題を解決するものであって、カスタマイズされたタスクモデルに新たに追加された部分タスクモデルを統合することが可能なインタフェースアーキテクチャを提供することを目的とするものである。また、本発明の他の目的は、操作やフィードバックに関するユーザの持つ知識を変更することなしに、新規追加モジュールの機能を利用することができ、ユーザのタスク要求のより多くの部分を解決できるインタフェースアーキテクチャを提供することである。さらに本発明の他の目的は、ユーザ及びインタフェースアーキテクチャが部分タスクモデルの一部を組み合わせた、変更しカスタマイズできるようにすることである。本発明の他の目的は、新規に追加された部分タスクモデルから既存の機能モジュールの機能が利用できるだけでなく、既存の部分タスクモデルあるいはカスタマイズされたタスクモデルから新規機能モジュールの機能が利用でき、既存の部分タスクモデルから新たに追加されたモジュールの実行時のコンテキストが参照できるようにすることである。

#### 【0070】

【課題を解決するための手段】そのために本発明は、独立性を持った要素機能に分割された複数のエージェントと共有媒体として種々の情報を読み書きするスタジオとを備えたアーキテクチャモデルにおいて、スタジオに送出されたタスク状態記述の履歴をコンテキスト情報として管理するスタジオ管理エージェントと、蓄積したコンテキスト情報を用いて契約ネットプロトコルにおける入札評価をする入札アービタエージェントと、コンテキスト依存処理の経過をユーザに説明し、コンテキスト依存処理をカスタマイズする手段をユーザに提供する対話マネージャエージェントとを有し、エージェント群がスタジオを介しスタジオ関数群を用いてタスク状態記述を授受し、スタジオ管理エージェントの管理するコンテキストに基づき入札アービタエージェントが入札を評価することにより、モジュール群が相互のコンテキストに応じて動作するように構成したことを特徴とするものである。

#### 【0071】

【作用】本発明に係るヒューマンインタフェースのアーキテクチャモデルでは、エージェント群がスタジオを介してタスク状態記述を授受するためのスタジオ関数群と、蓄積したコンテキスト情報を用いて契約ネットプロトコルにおける入札評価する入札アービタエージェントと、コンテキスト依存処理の経過をユーザに説明し、ユーザにコンテキスト依存処理をカスタマイズする手段を提供する対話マネージャエージェントとを有し、エージェント群がスタジオを介しスタジオ関数群を用いてタスク状態記述を授受し、スタジオ管理エージェントの管理するコンテキストに基づき入札アービタエージェントが入札を評価することにより、モジュール群が相互のコン

テキストに応じて動作するので、ユーザとシステムの間で合意の一部であるタスクモデル（システムの保有するユーザのタスク遂行に関する知識）に着目し、新たなモジュールの追加に際して、ユーザとシステムの間で合意を可能な限り維持しながら、新規追加モジュールの機能を利用する機構を提供することができる。

#### 【0072】

【実施例】以下、本発明の実施例を図面を参照しつつ説明する。図1は本発明に係るヒューマンインタフェースのアーキテクチャモデルのスタジオの構造の1実施例を示す図であり、100はスタジオ、101はユーザ層、102はタスク状態記述層、103はシングルキャスト層、104はブロードキャスト層、105は＜対話マネージャ＞エージェント、106は＜スタジオ管理＞エージェント、107は＜入札アービタ＞エージェント、108は一般のエージェント、201はファンクションレベル、202はユーザタスクレベル、203はセマンティックレベル、204はインタラクションレベルを示す。

【0073】図1において、スタジオ100は、ユーザ層101、タスク状態記述層102、シングルキャスト層103、ブロードキャスト層104の4つの層からなり、①エージェント群が相互に通信する手段を提供する、②エージェント群の通信履歴をコンテキスト情報として維持する、③契約ネットプロトコルを用いたコンテキスト依存処理を提供する、④コンテキスト依存処理をユーザに説明し、ユーザの変更を受け入れる、という機能を持つ。

【0074】スタジオ100の機能は、Lindaプログラミングモデル（米国：SCIENTIFIC Computing Associatesの製品）の提供するダブルスペース、および必須エージェント（System Agents）と呼ばれる特殊なエージェント群によって実現されている。必須エージェントには、図示のように＜対話マネージャ＞エージェント105、＜スタジオ管理＞エージェント106、＜入札アービタ＞エージェント107の3つがある。＜スタジオ管理＞エージェント106は、エージェント群の通信履歴をコンテキスト情報として維持する役割を持つ。＜入札アービタ＞エージェント107は、＜スタジオ管理＞エージェント106の蓄積したコンテキスト情報を利用して、入札の評価を行うことで、契約ネットプロトコルを用いたコンテキスト依存処理を実現する。＜対話マネージャ＞エージェント105は、コンテキスト依存処理のユーザへの説明、ユーザからのカスタマイズに重要な役割を果たす。そして、一般のエージェント群108がスタジオの機能を利用するためにスタジオ関数群が提供されている。

【0075】レイヤ構造のスタジオ100において、エージェント設計者は、タスク状態記述層102にのみ関心を払えばよい。一般のエージェント108は、スタジ

オ関数 (Studio Functions) を用いてスタジオのタスク状態記述層 102 にタスク状態記述 (Task-Phase Description; 以後 TPD と略) を読み書きすることで通信を行う。タスク状態記述層 102 に示す ex. 1 は、送出されるタスク状態記述の例を示したものである。このようにタスク状態記述は、3つの文字列と4つのポインタを情報として持つ。

【0076】シングルキャスト層 103、ブロードキャスト層 104 は、Linda のダブルスペースにより実現されており、タスク状態記述層 102 におけるエージェント間の通信は、スタジオ関数が関数内部で通信プリミティブ (communication primitives) を呼び、通信プリミティブが最終的に Linda プリミティブを用いることで、ブロードキャスト層 103、シングルキャスト層 104 へのダブルの授受に変換される。

【0077】シングルキャスト層 103 は、相手先のエージェントを指定した通信をサポートし、ブロードキャスト層 104 は、契約のアナウンスなど相手先を特定しないブロードキャスト通信をサポートする。例えば、タスク状態記述層 102 における ex. 1 のタスク状態記述の送信は、シングルキャスト層 103 では ex. 2 のようなダブルの送信となり、ブロードキャスト層 104 では ex. 3 のようなダブルの送信となる。

【0078】ユーザは、ユーザ層 101 にのみ関心を払えばよい。タスク状態記述層 102 におけるタスク状態記述は、ユーザ層 101 でユーザ可読な自然言語風のテキスト形式に変換されてユーザに提示される。例えばタスク状態記述層 102 における ex. 1 のタスク状態記述は、ユーザ層 101 における ex. 4 のようなテキストに変換される。

【0079】また、スタジオ 100 は、送出されるタスク状態記述の内容に基づき、ファンクションレベル 201、ユーザタスクレベル 202、セマンティックレベル 203、インタラクションレベル 204 の4つのレベルに分割されている。

【0080】次に、スタジオの持つ機能で、エージェント群が相互に通信する手段を提供する機能、エージェント間の通信履歴をコンテキスト情報として維持する機能の実現法について説明する。図2はタスク状態記述の構成例を示す図、図3はスタジオのレベルを説明するための図である。

【0081】まず、エージェント間の通信媒体でありコンテキスト情報の基本構築手段であるタスク状態記述 (TPD: Task Phase Description) について説明する。各エージェントは、スタジオ関数を用いて TPD をスタジオ 100 のタスク状態記述層 102 に読み書きすることで通信を行う。また、スタジオ 100 に送出された TPD の履歴をコンテキスト情報として利用する。TPD は、図2に示すように3つの文字列と2つのエージェントへのポインタ、2つの TPD へのポインタからな

り、コンテキスト依存処理を行うために、この7種類の情報を用いる。この情報のそれぞれの意味は以下のようなものである。

【0082】Task は、TPD により状態が記述されているタスクを表す文字列である。

【0083】Phase (Ph) はタスクの状態を示し、“Announced”、“Executing”、“Completed”、“Error”、“Underfined”のいずれかの文字列である。Announced

は、実行すべきタスクが存在するが、まだ実行されていない状態を示す。スタジオ関数 Metaphor Call(), Semantic Command() で送出される TPD がこの Phase を持つ。Executing は、タスクが Server 項で指示する部分タスクエージェントにより、Metaphor 項のメタファ環境エージェントを用いて遂行中である状態を示し、この部分タスクエージェントとメタファ環境エージェントは契約関係にある。Completed は、タスクが成功裡に終了した状態を示す。Error は、タスクの実行に失敗した状態を示す。Underfined は、タスクが未定である状態を表す。例えば Metaphor Announce() により送出される TPD

は、この Phase を持ち、この Phase を持つ TPD は、契約アナウンスとして送出される。

【0084】Metaphor Class (MC) は、そのタスク実行の外見として雇用されているメタファの種類を示す文字列であり、例えば “News Paper”、“TV-Program” 等の値を持つ。Function Class (FC) は、そのタスク実行の機能の種類を示す文字列であり、例えば “DB”、“Mail” 等の値を持つ。

【0085】Server (Sv) は、このタスクを実行しているエージェントの ID である。

【0086】Metaphor (Mp) は、タスクに雇用されていた、または現在雇用されているメタファ環境エージェントの ID である。

【0087】Super Task (ST) は、そのタスクの上位タスクを記述するスタジオ上の別の TPD の ID であり、TPD の ID は、その TPD を送出したエージェントの ID と、そのエージェント内で一意のメッセージ ID からなる整数値のペアである。

【0088】Causal Event (CE) は、そのタスクの生起する原因となったイベントを示す TPD の ID である。

【0089】次に、タスク状態記述をスタジオに送出するためのスタジオ関数群について説明する。スタジオ 100 は、送出される TPD の種類に対応して図3に示すようにインタラクションレベル 204、セマンティックレベル 203、ユーザタスクレベル 202、ファンクションレベル 201 からなる4つのレベルに分割されている。インタラクションレベル 204 では、TPD の種類として、Syntactic Event があり、TPD によりマウスクリック等のウインドウレベルの操作を表し、メタファ環境エージェント 300 が TPD を送出する。セマンテ

イックレベル203では、TPDの種類として、Semantic Event、Semantic Command、SemanticReplyがあり、TPDによりメタファ環境エージェント300と部分タスクエージェント500の間で対話を行い、メタファ環境エージェント300と部分タスクエージェント500の間でTPDの授受を行う。ユーザタスクレベル202では、TPDの種類として、Metaphor Announce、Metaphor Call、Execute Task、Complete Taskがあり、TPDによりメタファ環境エージェント300と部分タスクエージェント300の間の契約アナウンス及び部分タスクエージェント500の実行状態を表し、メタファ環境エージェント300と部分タスクエージェント500の間でTPDの授受を行う。ファンクションレベル201では、TPDの種類として、Function Command、Function Replyがあり、TPDによりファンクションエージェント400と部分タスクエージェント500の間の対話を行い、ファンクションエージェント400と部分タスクエージェント500の間でTPDの授受を行う。このようにTPDを送出するスタジオ関数として、TPDの種類に応じた10種の関数が用意されている。また、TPDを受信するためのスタジオ関数として、Wait Pattern()、Receive()、Wait Call()の3つの関数がある。

【0090】メタファ環境エージェントは、現実世界のオブジェクト（例えば新聞、テレビその他）に関するまとまった知識と、その知識をユーザに想起させるための表現を有する。これは、ユーザが現実世界の知識を用いてシステムの操作とフィードバックに関する合意を形成するのに役立つ。このようなメタファ環境エージェントの利用手続きをメタファプロトコルとして公開する。

【0091】メタファプロトコルは、＜新聞＞、＜TV番組＞といったメタファの種類（metaphorClass）毎に定義され、Syntactic Event、Semantic Event、Semantic Command、Semantic Replyのtask項の形で規定する。また、Syntactic Eventに対応してどのSemantic EventあるいはMetaphor Announceが発行されるかを規定する。

【0092】Syntactic Event、Semantic Eventは、ユーザのメタファ環境エージェントへの操作を表すTPDである。Syntactic Eventのtask項は、ユーザのメタファ環境エージェントに対して行った操作の内容を表し、Syntactic Eventのtask項は、Syntactic Eventのtask項に対して、部分タスクエージェントとの契約関係に基づき意味的な解釈を施し、契約関係にある部分タスクエージェントに伝達するものである。

【0093】図4はSyntactic Event、Semantic Eventの例を示す図であり、(1)の例では、「ユーザが新聞メタファの左4番目の記事をクリックした」ことを表し、このSyntactic Eventに対して、(2)の「ID3131の記事にユーザが注目した」ことを示すSemantic Eventが送出される。Semantic Eventは、契約のアナウンスとなる。

【0094】Semantic Command、Semantic Replyは、ユーザへの表示を変更するために、部分タスクエージェントが契約関係にあるメタファ環境エージェントに対して送出すべきTPDとその応答である。

【0095】ファンクションエージェントは、機能モジュールであり、その利用手続きをファンクションプロトコルとして公開する。ファンクションプロトコルは、＜DB＞、＜Mail＞といった機能の種類（functionClass）毎に定義され、Function CommandとFunction Replyのtask項の形式を規定する。Function CommandとFunction Replyは、ファンクションエージェントの機能を利用するために部分タスクエージェントが送出すべきTPDとその応答である。

【0096】タスクを遂行中の部分タスクエージェントは、ユーザに対してタスクの遂行状態を提示するためのメタファ環境エージェントを1つ必要とする。これら2つのエージェントは契約関係にあるという。

【0097】Execute Task、Complete Taskは、server項が指示する部分タスクエージェントが、metaphor項が指示するメタファ環境エージェントを用いてtask項のタスクを実行中である、あるいは終了したことを示す。

【0098】図5はExecute Taskの例を示す図である。このTPD(1)は、「Read NewsArticle ID3131」というタスクが「Read 10/21's News」のサブタスクとして、「Focus NewsArticle ID3131」が原因で起動され、契約関係にある2つのエージェントにより現在実行中であることを示している。Execute TaskのMetaphor項は、server項の部分タスクエージェントと契約関係になるメタファ環境エージェントを指示する。

【0099】Metaphor Announce、Metaphor Callは、契約関係を新たに結ぶためのTPDである。Metaphor Announceは、ユーザが、メタファ環境エージェントを指定したときに契約のアナウンスとして発行される。MetaphorClass項は、メタファプロトコルの種類を指し、このメタファプロトコルに対応可能な部分タスクエージェント全てが入札に参加できる。スタジオに実装された入札評価手続きにより、現在のコンテキストに最も適切なコンテキスト条件を入札した部分タスクエージェントが選択され、Metaphor Announceを発行したメタファ環境エージェントとの間で契約関係が結ばれる。

【0100】Metaphor Callは、実行すべきタスクを有する部分タスクエージェントが、MetaphorClass項で指定される種類のメタファ環境エージェントと契約関係を結ぶために用いる。task項には、契約関係成立後に実行するタスクを記述する。

【0101】図6は部分タスクエージェント群の動作を説明するための図である。部分タスクモデル中のタスク遂行のコンテキストに依存した条件分岐のうち、カスタマイズ可能な分岐を契約ネットプロトコルの入札機構により実現する、というアプローチにより、各ベンダの設

計する部分タスクモデルは、複数の部分タスクエージェント内に分割して実現される。

【0102】各部分タスクエージェントは、契約関係に基づいたタスク遂行形式の記述と、契約アナウンスに対する入札の形式の記述からなる2種類の記述を部分タスクモデルとして持つ。

【0103】まず、契約関係に基づいたタスク遂行形式の記述は、

Semantic Event/Reply→

(コンテキスト条件×内部状態→ (Execute Task+Complete Task +Function Command +Semantic Command)

\*)

となり、この部分タスクモデルは、契約関係のあるメタファ環境エージェントからSemantic EventあるいはReplyを受け取った時に、コンテキスト条件及び内部状態に応じていかなるTPDの列を発行するかを規定する。

【0104】契約アナウンスに対する入札の形式の記述は、

Semantic Event→

(コンテキスト条件×Execute Task)

Metaphor Announce →

(コンテキスト条件×Execute Task)

となり、この部分タスクモデルは、契約のアナウンスとなるSemantic Eventと、Metaphor Announce に対して、いかなるコンテキスト条件で入札を行い、落札時にいかなるタスクを実行するかを記述する。

【0105】ベンダは、ユーザのモジュール群をいかに使うかを想定すると同時に、モジュール群がいかに使われるべきかを想定している。契約関係に基づくタスク遂行形式の記述には、ベンダが「こう使うべきであり変更不可」と想定したタスク遂行がモデル化されている。一方、契約アナウンスに対する入札の形式の記述には、ユーザまたはシステムによりカスタマイズ可能とベンダが想定したタスク遂行がモデル化されている。 \*

\* 【0106】入札のコンテキスト条件は、ユーザまたはシステムによりカスタマイズ可能である。また、メタファプロトコルの一部として、契約のアナウンスが公開されているため、他のベンダが入札を行うエージェントを設計することができる。

【0107】これらの部分タスクモデルが修正され、追加されることにより、部分タスクエージェント群が全体としてタスクモデルを形成しユーザのタスク遂行のコンテキストに応じた処理を可能にしていると見ることが出来る。部分タスクエージェント群は、全体として図6のように動作する。メタファ環境エージェントがユーザの操作に応じて発行したSemantic Eventを受け取り(1)、タスクモデル(A)により現在のコンテキストで最も適切な動作を判断し、以下のいずれかの動作(またはその組み合わせ)を実行する。

【0108】Function Commandによりファンクションエージェントの機能を実行する(2)。Semantic Commandを発行し、ユーザに対する表示の変更を行う(3)。

【0109】Execute Task/Complete Taskにより、自身の行っているタスクの実行状態をスタジオに表明する(4)。

【0110】次に、タスク状態記述の履歴をコンテキスト情報として蓄積する<スタジオ管理>エージェントの役割について説明する。発行されたTPD間の相互関係は、コンテキスト情報として維持する必要があり、<スタジオ管理>エージェントがこのコンテキスト情報の管理を行う必須エージェントである。

【0111】スタジオ関数によるTPDの発行は、最終的には通信プリミティブinform(), sendto(), contract()によりシングルキャスト層へLindaのタプルとして送出される。シングルキャスト層に送出されるタプルの一般的な形式は次のようになる。

【0112】

Receiver	Com	rid	sid	mesno	TPDの情報
----------	-----	-----	-----	-------	--------

Receiverは、このタプルを受け取るエージェントのIDであり、スタジオ関数から呼ばれる通信プリミティブによりシングルキャスト層に送出されるタプルでは、この項に常に0が指定される。つまり、発行される全てのTPDは、まず、ID=0の<スタジオ管理>エージェントが受信することになる。<スタジオ管理>エージェントは、受信したタプルをコンテキスト情報として蓄積した後、タプルのCommunication typeに基づきタプルを他のエージェントに転送する。

【0113】Communication type (com)は、このタプルの通信形態を示し、“Inform”、“Sendto”、“Contract”の値を持ち、これらは通信プリミティブの種類に対応する。

【0114】receiver-id (rid) は、通信形態が“Sendto”のときのみ意味を持ち、タプルの最終的な受取先となるエージェントのIDを示す。

【0115】sender-id (sid), message-no (mesno) の2項により、TPDは一意に特定される。sender-idは、通信プリミティブを実行したエージェントのID、message-noは、そのエージェント内で一意に決まる発行番号である。

【0116】上記のように<スタジオ管理>エージェントは、読み込んだTPD情報をタイムスタンプと付加情報(com, rid, sid, mesno)とともに、TPD履歴情報として蓄積する。本発明では、この履歴をコンテキスト情報として<入札アービタ>エージェントが契約の入札

を評価する際に用いる。

【0117】次に、入札の形式とコンテキスト情報を利用した評価手続きについて説明する。図7は入札評価の例を示す図である。

【0118】入札の評価の基本的な方針は、契約のアナウンス時のコンテキストに最も適切な入札者に落札させるというものであり、入札の評価手続きは、必須エージェント<入札アービタ>により実装される。

【0119】契約の入札は、

〔プラン〕〔コンテキスト条件列〕の形式で行われる。プランは、契約に成功したとき実行するタスクを示す。コンテキスト条件列は、コンテキスト条件の列であり、コンテキスト条件は、以下の条件式をandで結合したものである。同一のTPDに対して、andで結合された条件式が全て成功したときに、コンテ\*

{ [Plan 1] {

[Task== "A" and Phase== "Completed"]

[Task== "B" and Phase== "Completed"]

}

のような入札は、「Aというタスクが終了しており、それ以前にBというタスクが終了していれば、Plan1を私が実行するのが適している」という入札者の主張を示す。

【0121】あるコンテキストでは、コンテキスト条件が成立する入札が複数ある可能性がある。これらの入札から、最もコンテキストにspecificな入札を選択するための評価手続きが必要である。入札の評価は、<入札アービタ>エージェントにより評価されるが、これは、入札の評価が複数の入札評価ルールで行われるより、単一の入札評価ルールで行われる方がユーザにとって理解しやすいという理由による。

【0122】本発明の入札評価ルールは、以下のように単純なものである。<スタジオ管理>エージェントは、TPDを時系列で蓄積している。

【0123】入札の評価は、各入札のコンテキスト条件列と、<スタジオ管理>エージェントの管理するTPD履歴とのパタンマッチングにより行われる。図7に示す入札評価の例のように、TPD履歴を過去に遡り、各コンテキスト条件列の要素とのパタンマッチングを行い、最初に全てのコンテキスト条件がマッチした入札を行ったエージェントを落札する。この場合、最初に2つのコンテキスト条件がマッチしたプラン1が落札する。

【0124】次に、本発明を具体的な例題により説明する。「新聞のような外見で情報を検索するツール（ツールAとする）」が存在する環境がある。ツールAの設計者は、ツールを起動したときに、起動時のコンテキストにより次の2つの動作に分かれるように設計した。

【0125】もし「今日、このツールを既に一度起動していた」ならば、→前回終了時の状態のままの新聞を表示する。

\* クスト条件がそのTPDにマッチしたという。

【0120】always — 常に成功

never — 常に失敗

TPD項==引数 — 引数とTPD項が一致すれば成功

TPD項!=引数 — 引数とTPD項が一致しなければ成功

TIMESTAMP <ddhhmmss — TIMESTAMPが引数より古ければ成功

10 TIMESTAMP >ddhhmmss — TIMESTAMPが引数より新しければ成功

コンテキスト条件列は、各コンテキスト条件の時間的な順序関係を表す列であり、現在から過去にさかのぼる方向に記述される。例えば

20 【0126】もし「今日、このツールを起動するのが初めて」ならば、→今日付の新しい情報を用いて新聞紙面を構成し直し、最初のページから表示する。

【0127】この環境に、新しく「ニュース番組のような外見で情報を検索するツール（ツールBとする）」が加わったとする。この場合、たとえツールA/ツールBが異なる設計者により作られたものであったとしても、ユーザはシステムを一体とみなすだろう。よって、ツールAの起動時に、次のような動作を期待するユーザがいることが考えられる。

30 【0128】もし「直前に、ツールBを使っていた」ならば、→ツールBで直前に見ていた情報を含む新聞ページを表示する。

【0129】コンテキストに依存した条件分岐を契約ネットワークプロトコルで行い、契約アナウンスおよび入札の形式を公開することで、設計者が現在のユーザのタスク遂行のコンテキストに沿って、新たな機能をユーザに提示するエージェントを設計、追加することができる。

【0130】新聞メタファ環境設計者は、次の4つのエージェントを提供した。

40 【0131】<新聞メタファ>：メタファ環境エージェント

<新聞で新しく今日のニュースを見る>：部分タスクエージェント

<新聞で引続き×月×日のニュースを見る>：部分タスクエージェント

<DB>：ファンクションエージェント

ユーザが新聞アイコンをクリックしたときのMetaphor Announceは、新聞メタファプロトコルの一部として公開されており、以下の形式をしている。

50 【0132】Task= "",Phase= "Undefined",MetaphorC

lass= "NewsPaper"

\*を見る」が、

部分タスクエージェント「新聞で新しく今日のニュース＊

〔〔新聞で 新しく 今日の ニュースを 見る〕；プラン

〔time<当日の0時〕；コンテキスト条件

〕〕

のように、日付更新をサーチする入札を行い、部分タスク＊見る＞が、

クエーエージェント＜新聞で引続き×月×日のニュースを見※

〔〔新聞で 引続き ×月×日の ニュースを 見る〕；プラン

〔〔Task == "Open NewsPaper" and

Phase == "Completed" and

MClass== "NewsPaper" and

〕

；コンテキスト条件

〕〕

のように、以前に発行した新聞オープン命令をサーチする入札を行うことで、日付の更新以後に新聞で今日のニュースを見ていれば、その続きから読むことができ、そうでなければ、新しく＜DB＞から検索した当日のニュースを読むことができる。

【0133】番組メタファ環境（ツールBに相当）の設☆

〔〔新聞で TVで見ていた ニュースを 見る〕；プラン

〔〔Task == "Open TV-Program" and

Phase == "Completed" and

MClass== "TV-Program" 〕

；コンテキスト条件

〕〕

これにより、日付の更新や、新聞オープンよりも最近に“TV-Program”が開いているというコンテキストの元では、新たに追加されたエージェント＜新聞でTVで見ていたニュースを見る＞の入札が落札することになる。

【0135】次に、Excuse-Customize処理について説明する。図8はシステムとユーザとのコンテキストの不一致表明画面の例を示す図、図9は弁解アイコンの生成画面の例を示す図、図10はユーザによる選択画面の例を示す図、図11は弁解ウインドウによるコンテキスト依存処理の理由説明画面の例を示す図、図12はカスタマイズウインドウによるカスタマイズの開始画面の例を示す図、図13は実演モード画面の例を示す図、図14はコンテキスト条件強化画面の例を示す図、図15はコンテキスト条件の緩和画面の例を示す図である。

【0136】ユーザの想定したコンテキストとベンダの想定したコンテキストが異なることがしばしば問題になる。このような場合、Excuse-Customize処理によれば、複数のエージェント間にわたるコンテキスト依存処理をユーザの可読な形で提示し、ユーザが変更できる。Excuse-Customize処理では、コンテキストの不一致が再現しないように、不一致が起こったとき、ユーザがシステムに対して割り込みボタンによる意思表示を行う。また、必要に応じてシステムに正しいコンテキストを教示することで、ユーザの独自のタスク遂行スタイルを構築していくことができる。

【0137】ユーザは、一度ウインドウで読んだ記事を

☆計者は、＜番組メタファ＞を追加するとともに、部分タスクエージェント＜新聞でTVで見ていたニュースを見る＞を以下のように設計することで、新聞アイコンのクリック時の動作を変更することができる。

【0134】

もう一度クリックして、その記事をスクラップブックにスクラップする機能を起動しようとした。しかし、システムは、単にウインドウをもう一度開いただけだった。このケースの例を示したのが図8である。

【0138】①. システムが予想外の動作をしたとき、ユーザは、割り込みボタンにより、異議を呈することができる。システムは、音声によりとりあえずユーザに対して謝っておく。

【0139】②. システムは、競合が生じた時点までバックトラックし、図9に示すようにExcuseアイコンを生成する。図9では、左下の「弁解」と書かれたアイコンがExcuseアイコンである。

【0140】③. ユーザは、Excuseアイコンをそのまま放置してかまわない。この場合、再びユーザが新聞アイコンをクリックした場合には、図10に示すように全ての可能な選択肢がユーザに提示され、ユーザにプランの選択を要求する。

【0141】④ユーザがExcuseアイコンをクリックすると、システムは、図11に示すようにExcuseウインドウを提示する。Excuseウインドウは、ユーザの操作から推論したコンテキストを競合解消の理由として提示し、可能な選択の一覧を提示する。

【0142】⑤. 可能な選択から、例えば、

〔スクラップブックに、id0004の記事をスクラップする〕

を選択すると、図12に示すようにカスタマイズウイン

ドウが開かれ、カスタマイズが可能になる。ここで、操作を特殊化する選択肢を選んだとき、システムは、図13に示すような実演モードになり、ユーザに具体的な操作を行うことを要求する。例えば、

〔スクラップブックに、id0004の記事をスクラップする〕

が起動するときには、

〔新聞の左ページの0番目の記事をバックグラウンドヘドラッグした〕

という操作が必要のように修正できる。

〔0143〕起動条件を強くする選択肢を選んだとき、ユーザは、図14に示すよう過去のコンテキストから、コンテキスト条件を追加し、その動作を起きにくくすることでカスタマイズを行うことができる。例えばスクラップブックの設計者は、ユーザが既に読んだ記事をもう一度クリックしたとき、

〔スクラップブックに \* の記事をスクラップする〕を起動されるように入札設計した。しかし、

〔ウインドウで \* の記事を読む〕

「いつでも」

の設計者がコンテキスト条件としてalwaysを設定してあった場合、ユーザがカスタマイズしない限り、新たな機能「スクラップブック」は起動されない。ユーザは、

〔ウインドウで \* の記事を読む〕

のコンテキスト条件を強化することで、スクラップブックを起動することができる。タスク状態履歴から、タスク

#### \* ク状態記述

〔新聞をレイアウトした〕

を取り出し、

〔ウインドウで \* の記事を読む〕

のコンテキスト条件を加えることで、一度レイアウトされた新聞に対して、2度、記事がクリックされたときは、スクラップブックが起動されるようにカスタマイズできる。

〔0144〕起動条件を弱くする選択肢を選んだとき、図15に示すようにユーザは、コンテキスト条件を削除し、望まれる動作を起しやすくすることができる。

〔0145〕次に、Excuse-Customize処理の実装設計例を説明する。図16はコンテキスト依存処理の理由説明画面の例を示す図である。

〔0146〕ユーザに対してコンテキスト依存処理を説明し、ユーザがコンテキスト依存処理をカスタマイズするための機構は、＜対話マネージャ＞エージェントの持つTPDをユーザ可読形式へ変換する機能、および＜スタジオ管理＞エージェントの蓄積したTPD履歴を用いて実現する。

〔0147〕＜対話マネージャ＞エージェントは、タスク状態記述層の情報をユーザ層の情報へ変換する。例えばMetaphor Class= "NewsPaper"のTPDに対して、次のような変換が行われる。

〔0148〕

Task	Phase	ユーザ可読形式
"Click Left Page/3rd Article"	"Completed"	新聞の左ページの3番目の記事をクリックした
"Go Page 3"	"Announced"	新聞を3ページにせよ
"Focus New Article ID3131"	"Completed"	新聞で記事ID3131に着目した
"Read Today's News"	"Announced"	新聞で今日のニュースを見る

のような文字列として提示される。

〔0149〕また、ユーザへのコンテキスト依存処理の理由説明は、図16に示すように＜スタジオ管理＞エージェントの蓄積したTPD履歴のうち、落札した入札のコンテキスト条件にマッチしたTPDの列、該当する契約アナウンスのCausal EventとなったTPDの列、および実際に落札し実行されたプランから作成される。

〔0150〕

〔発明の効果〕以上の説明から明らかなように、本発明によれば、エージェント群がスタジオを介してタスク状

態記述を授受するためのスタジオ関数群と、スタジオに送出されたタスク状態記述の履歴をコンテキスト情報として管理するスタジオ管理エージェントと、蓄積したコンテキスト情報を用いて契約ネットプロトコルにおける入札評価する入札アービタエージェントと、コンテキスト依存処理の経過をユーザに説明し、ユーザにコンテキスト依存処理をカスタマイズする手段を提供する対話マネージャエージェントとを有し、新たなエージェントが契約の入札に参加し、スタジオ管理エージェントの管理するコンテキストに基づき入札アービタエージェントが

入札を評価することにより、モジュール群が相互のコンテキストに応じて動作するので、ユーザとシステムとの合意の一部であるタスクモデル（システムの保有するユーザのタスク遂行に関する知識）に着目し、新たなモジュールの追加に際して、ユーザとシステムとの合意を可能な限り維持しながら、新規追加モジュールの機能を利用する機構を提供することができる。そのため、カスタマイズされたタスクモデルに新たに追加された部分タスクモデルを統合することが可能になり、操作やフィードバックに関するユーザの持つ知識を変更することなしに、新規追加モジュールの機能を利用することができ、ユーザのタスク要求のより多くの部分を解決できる。しかも、ユーザ及びインタフェースアーキテクチャが部分タスクモデルの一部を組み合わせたか、変更しカスタマイズでき、新規に追加された部分タスクモデルから既存の機能モジュールの機能が利用できるだけでなく、既存の部分タスクモデルあるいはカスタマイズされたタスクモデルから新規機能モジュールの機能が利用できる。既存の部分タスクモデルから新たに追加されたモジュールの実行時のコンテキストが参照できる。

#### 【図面の簡単な説明】

【図1】 本発明に係るヒューマンインタフェースのアーキテクチャモデルのスタジオの構造の1実施例を示す図である。

【図2】 タスク状態記述の構成例を示す図である。

【図3】 スタジオのレベルを説明するための図である。

【図4】 Syntactic Event、Semantic Eventの例を示す図である。

【図5】 Execute Taskの例を示す図である。

【図6】 部分タスクエージェント群の動作を説明するための図である。

【図7】 入札評価の例を示す図である。

【図8】 システムとユーザとのコンテキストの不一致表明画面の例を示す図である。

【図9】 弁解アイコンの生成画面の例を示す図である。

【図10】 ユーザによる選択画面の例を示す図である。

【図11】 弁解ウインドウによるコンテキスト依存処理の理由説明画面の例を示す図である。

【図12】 カスタマイズウインドウによるカスタマイズの開始画面の例を示す図である。

【図13】 実演モード画面の例を示す図である。

【図14】 コンテキスト条件強化画面の例を示す図である。

ある。

【図15】 コンテキスト条件の緩和画面の例を示す図である。

【図16】 コンテキスト依存処理の理由説明画面の例を示す図である。

【図17】 部分タスクモデルのカスタマイズによる合意の形成を説明するための図である。

【図18】 新しいモジュールの追加に伴う合意の破壊を説明するための図である。

【図19】 HP SoftBench環境の問題点を説明するための図である。

【図20】 先行発明のヒューマンインタフェースのアーキテクチャモデルの概念図である。

【図21】 先行発明に係るヒューマンインタフェースのアーキテクチャモデルの1実施例を示す図である。

【図22】 先行発明に係るヒューマンインタフェースのアーキテクチャモデルの動作を説明するための図である。

【図23】 先行発明に係るヒューマンインタフェースのアーキテクチャモデルの動作を説明するための図である。

【図24】 先行発明に係るヒューマンインタフェースのアーキテクチャモデルの動作を説明するための図である。

【図25】 先行発明に係るヒューマンインタフェースのアーキテクチャモデルの他の実施例を示す図である。

【図26】 先行発明に係るヒューマンインタフェースのアーキテクチャモデルの動作を説明するための図である。

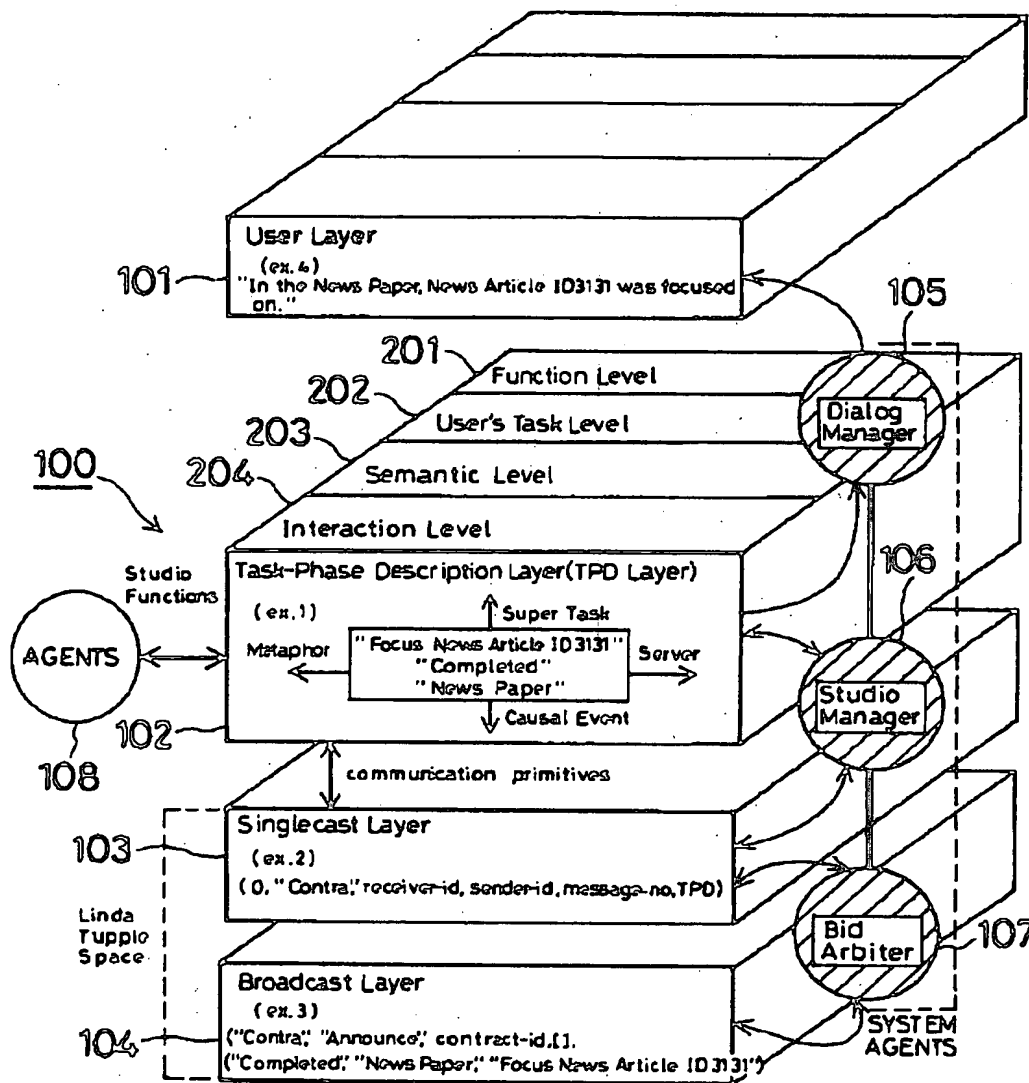
【図27】 先行発明に係るヒューマンインタフェースのアーキテクチャモデルの動作を説明するための図である。

【図28】 先行発明に係るヒューマンインタフェースのアーキテクチャモデルの動作を説明するための図である。

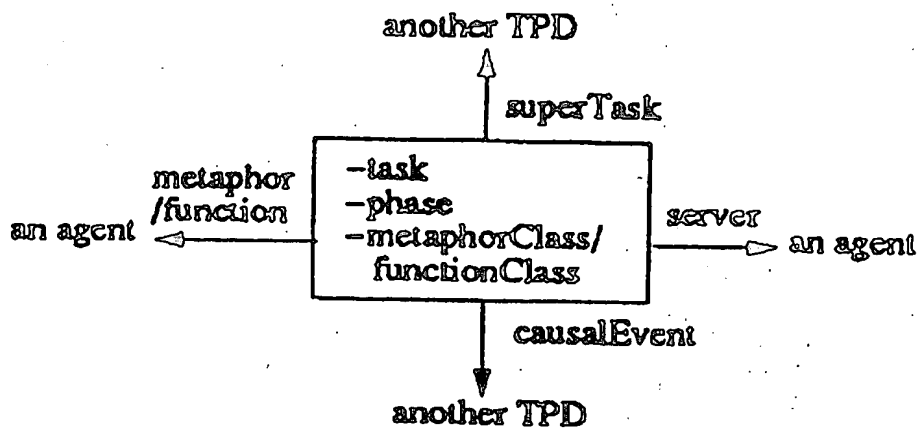
#### 【符号の説明】

100…スタジオ、101…ユーザ層、102…タスク状態記述層、103…シングルキャスト層、104…ブロードキャスト層、105…必須エージェント<対話マネージャ>、106…必須エージェント<スタジオ管理>、107…必須エージェント<入札アービタ>、108…一般のエージェント、201…ファンクションレベル、202…ユーザタスクレベル、203…セマンティックレベル、204…インタラクションレベル

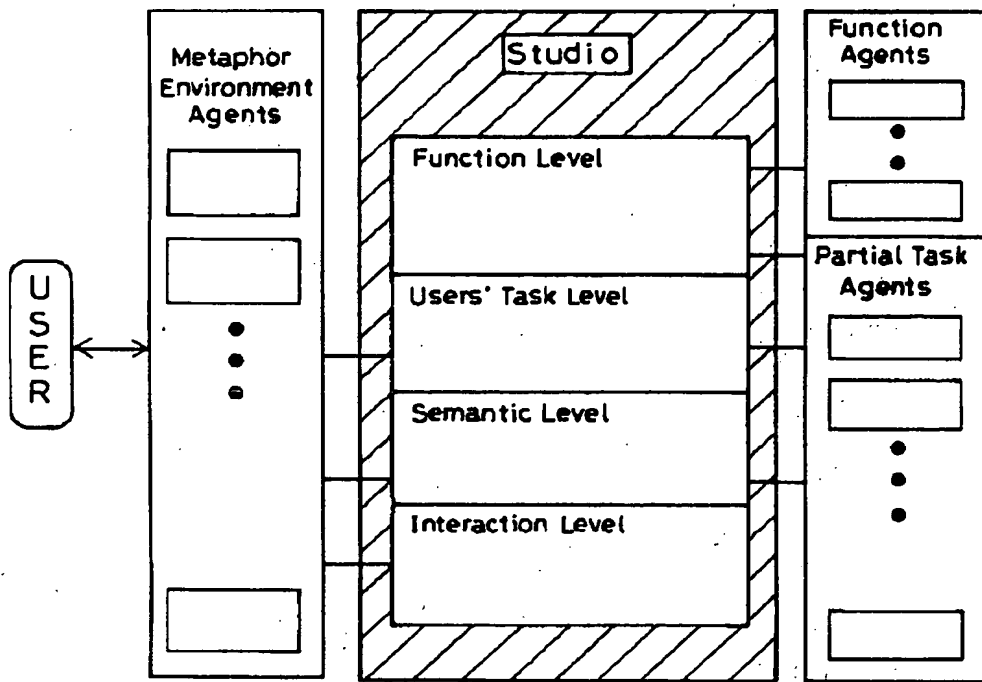
【図1】



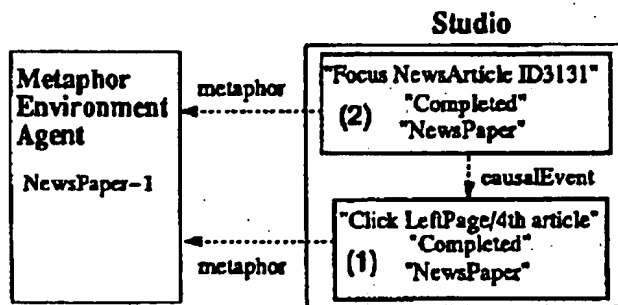
【図2】



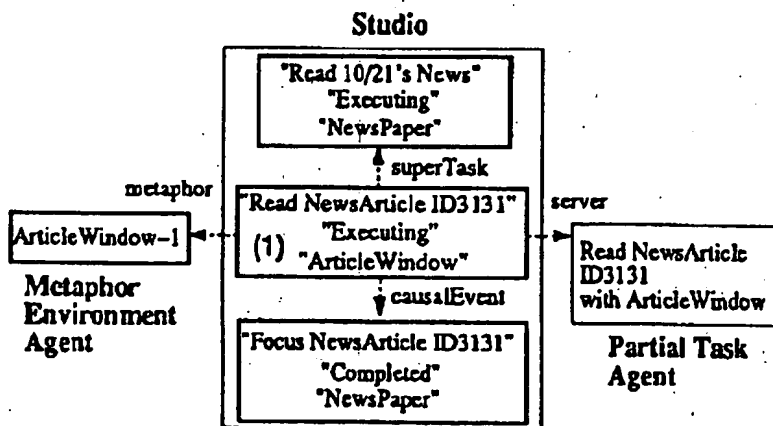
【図3】



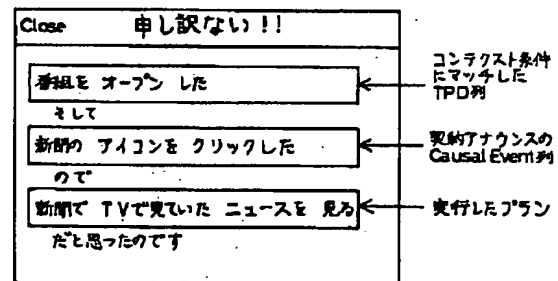
【図4】



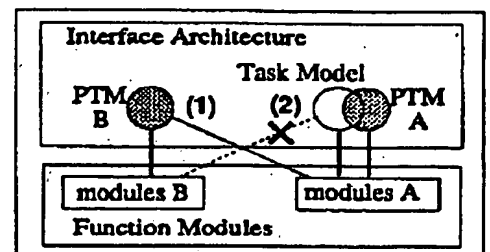
【図5】



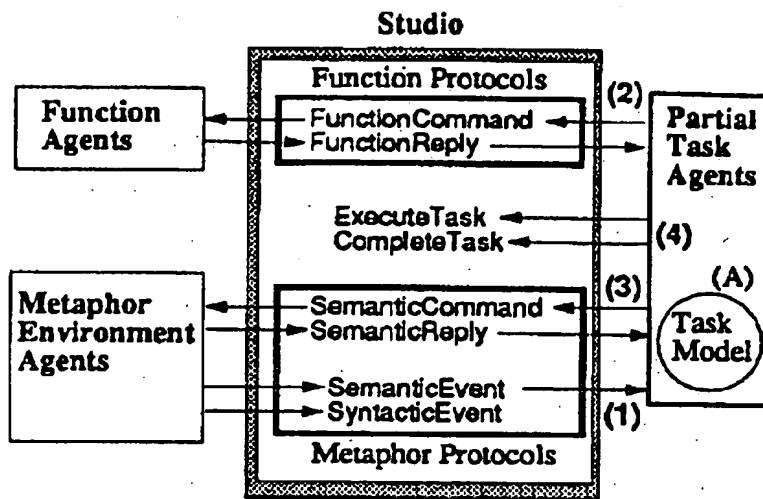
【図16】



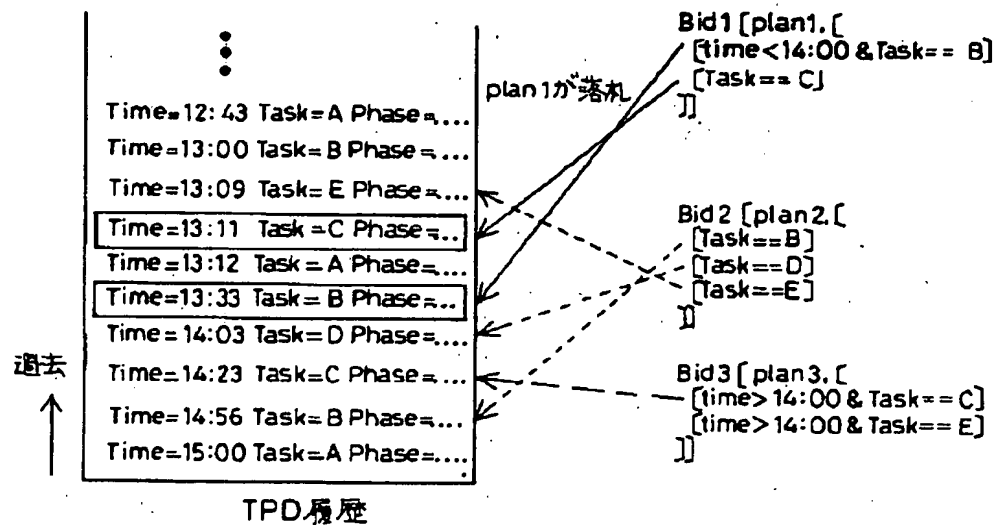
【図19】



【図6】



【図7】



【図8】

Figure 8 is a schematic diagram of a document layout. It consists of two main rectangular frames arranged side-by-side. The left frame is divided into four horizontal sections, each containing a grid of vertical lines. The right frame is also divided into four horizontal sections, but the top section is a solid black rectangle, and the bottom three sections contain a grid of vertical lines. Above each frame is a header area with text. Below the frames is a large black rectangular area.

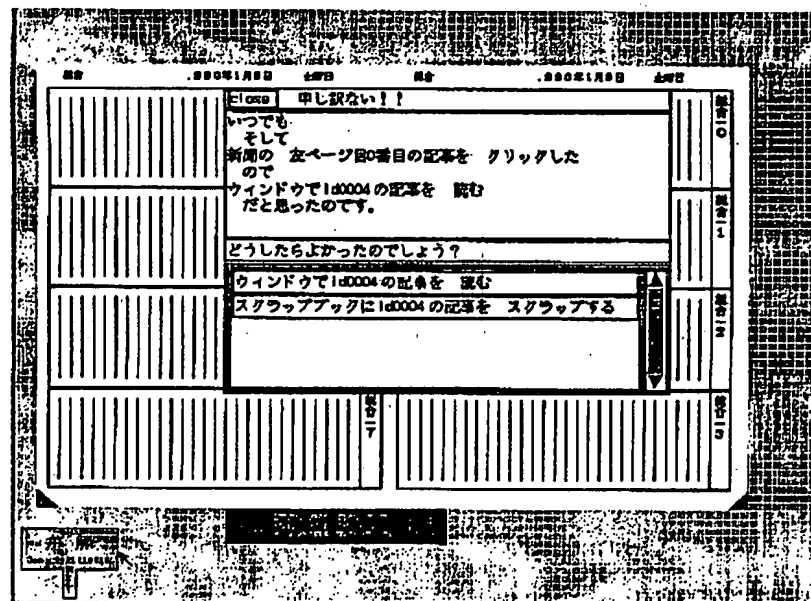
【図9】

Figure 9 is a schematic diagram of a document layout, similar to Figure 8. It consists of two main rectangular frames arranged side-by-side. The left frame is divided into four horizontal sections, each containing a grid of vertical lines. The right frame is also divided into four horizontal sections, each containing a grid of vertical lines. Above each frame is a header area with text. Below the frames is a large black rectangular area. In the bottom left corner, there is a small box with the text "弁 別" and "No. 9 00 10 0720".

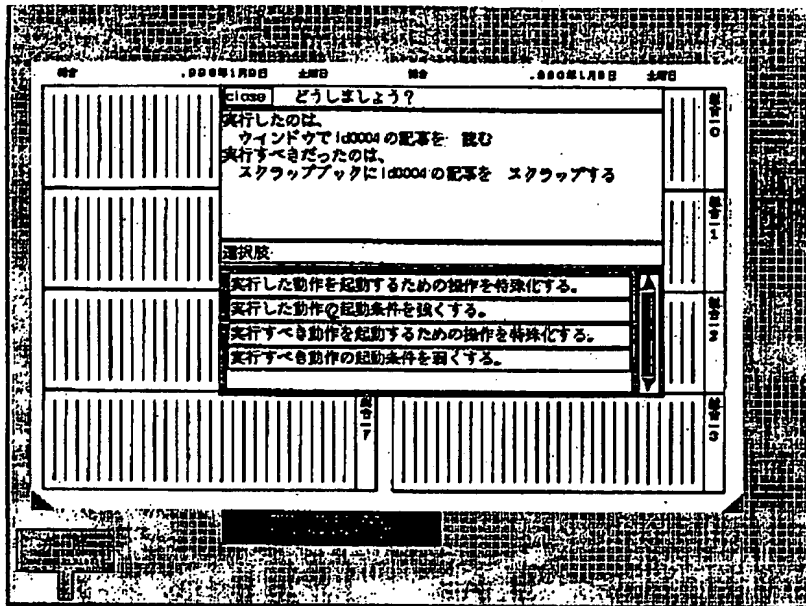
【図 10】



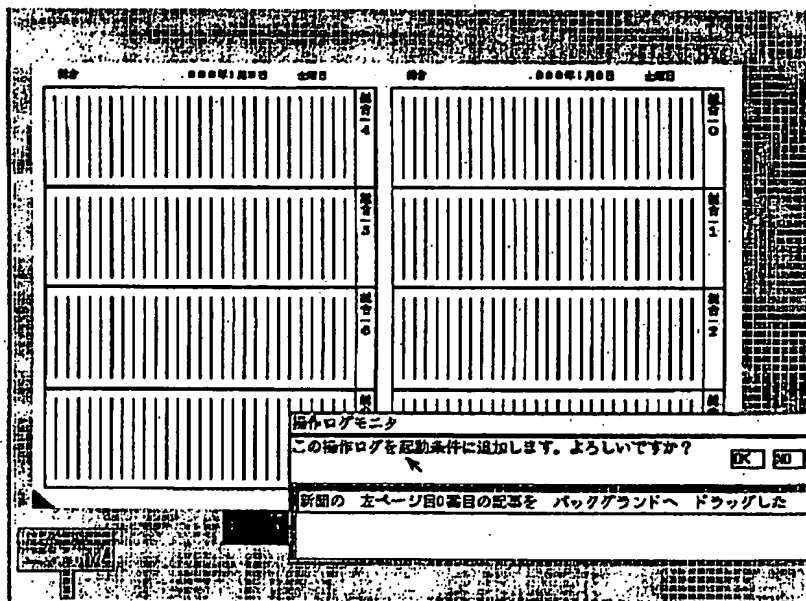
【図 1 1】



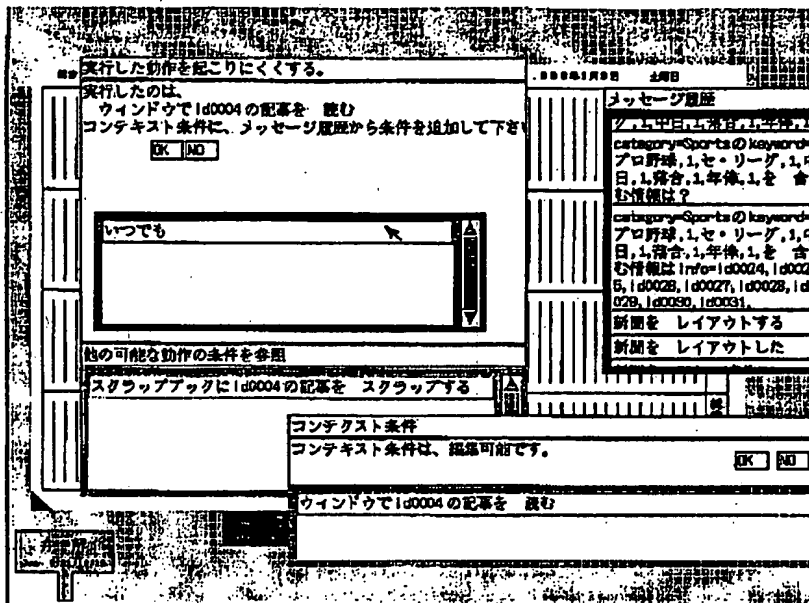
【図12】



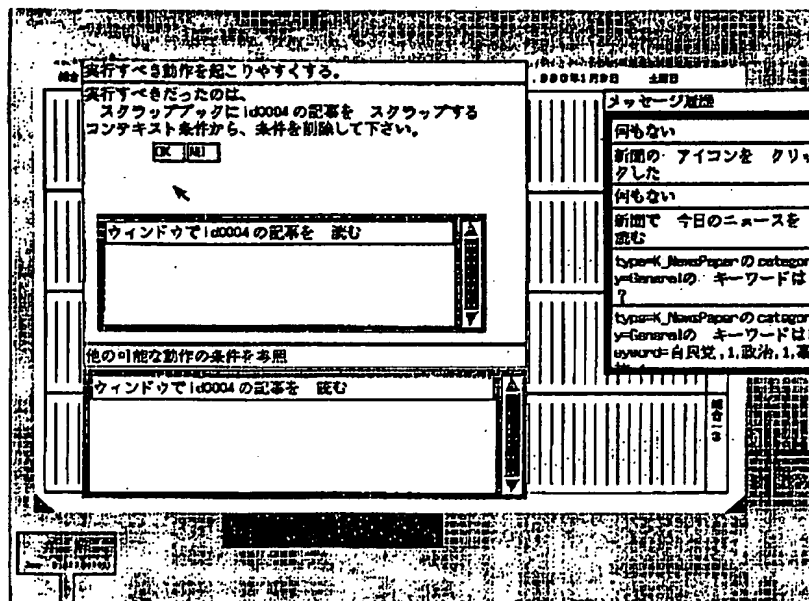
【図13】



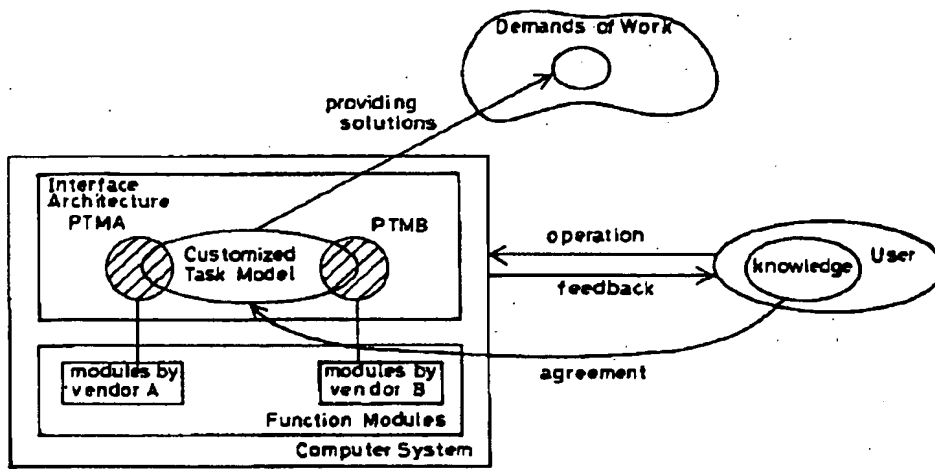
【図14】



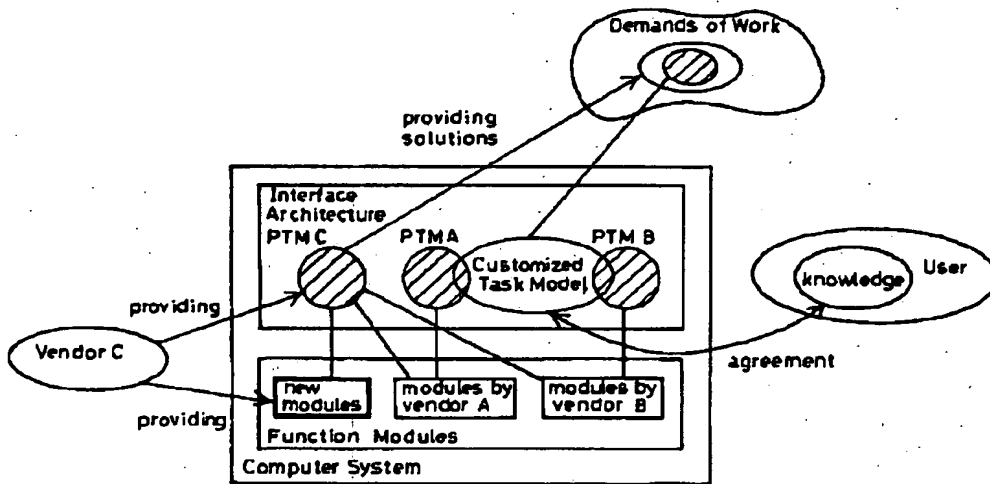
【図15】



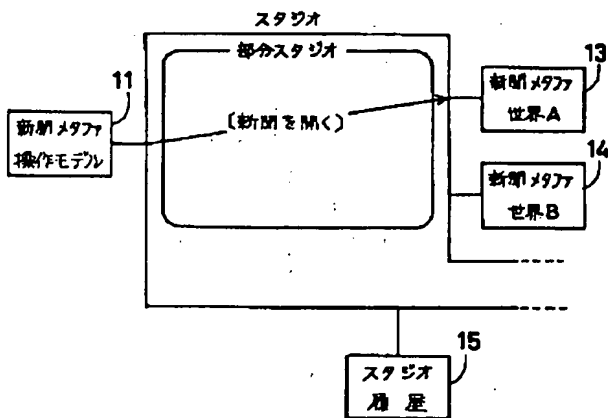
【図 17】



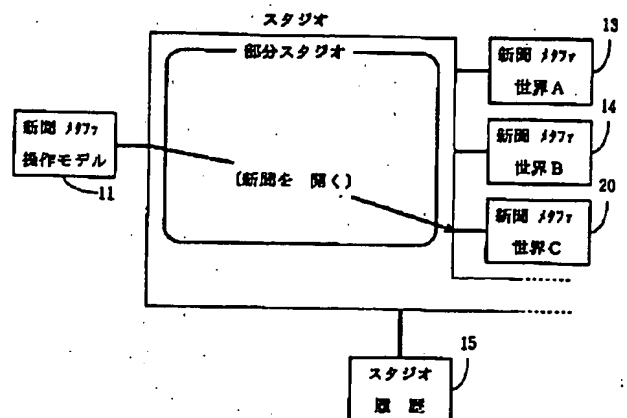
【图 18】



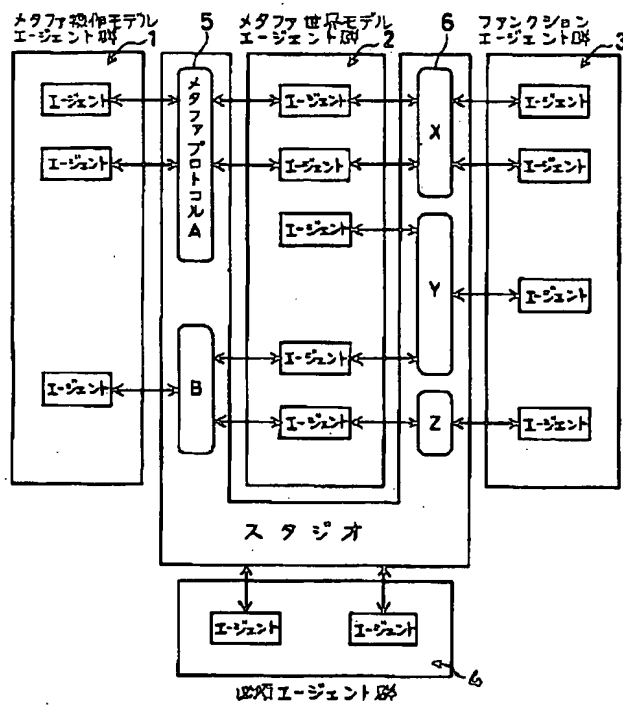
【図 2 4】



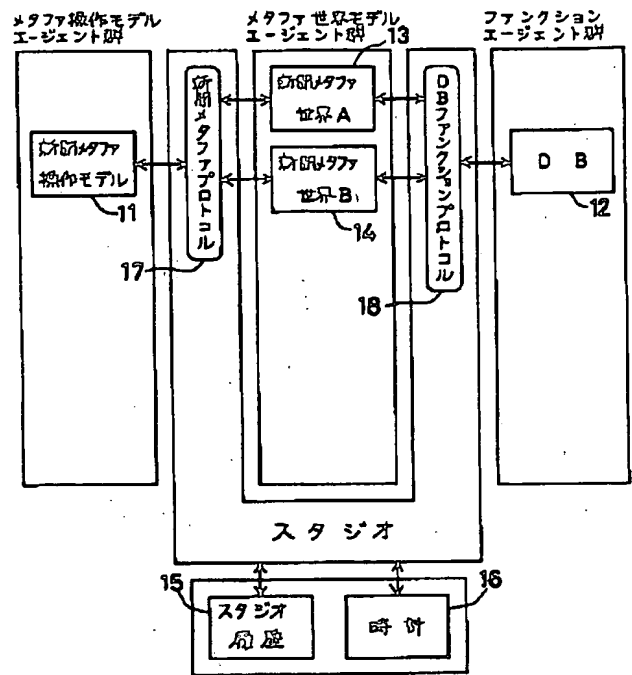
【图 28】



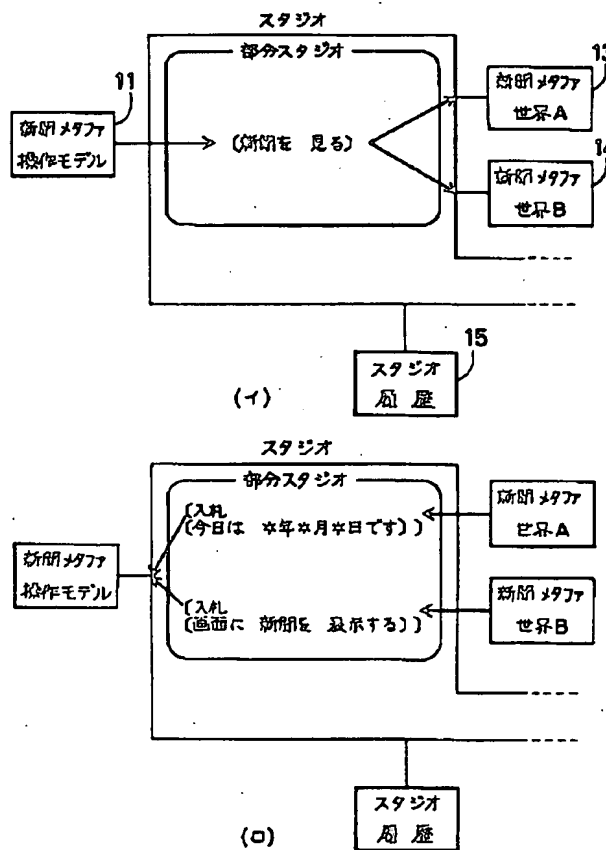
【図20】



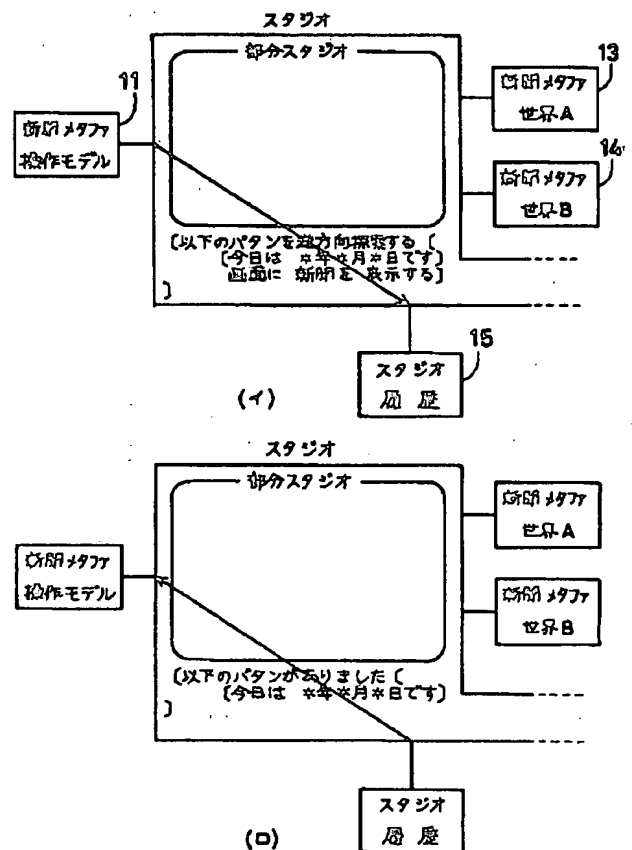
【図21】



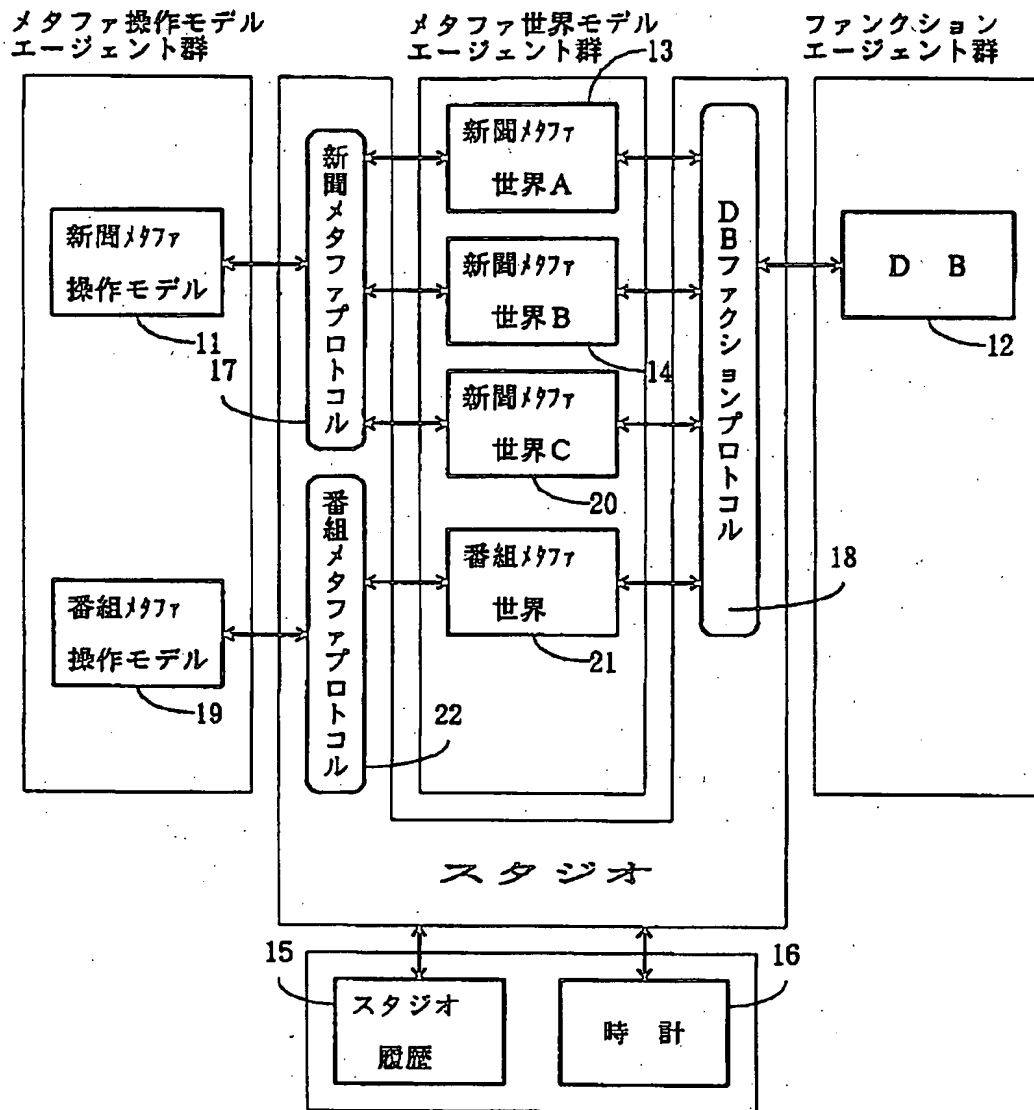
【図22】



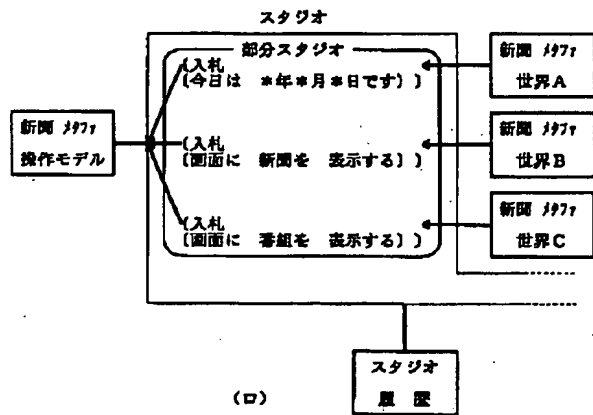
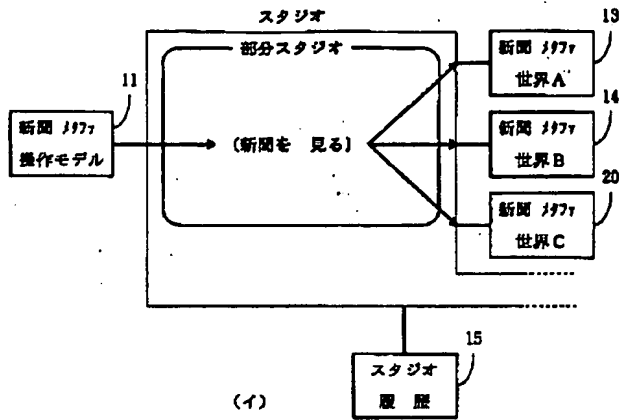
【図23】



【図25】



【図26】



【図27】

